

COMPRESSOR LOAD STAND:
COMMISSIONING AND CONTROL STRATEGIES

ANDREW E. CAUSEY
ME597 RESEARCH PROJECT
MAY 1998

19981119 016

TABLE OF CONTENTS

LIST OF FIGURES.....	4
2. CYCLE DESCRIPTION.....	5
3. HARDWARE SETUP AND DESCRIPTION.....	8
3.1 MEASURING DEVICES	9
3.2 ELECTRICAL SYSTEM.....	10
3.3 LOAD STAND CABINET DESCRIPTION.....	10
3.4 CONDENSER	11
3.5 ELECTRONIC EXPANSION VALVES	11
3.6 COMPRESSOR	11
3.7 SAFETY DEVICES	12
3.8 REFRIGERANT PIPING ACCESSORIES.....	12
3.9 SOLENOID VALVE.....	12
3.10 ELECTRIC RELAYS	13
4. DATA ACQUISITION SYSTEM.....	13
5. CONTROLLER DESCRIPTION	15
6. CONTROLLER PROGRAMMING	16
6.1 ANALOG INPUTS (AI).....	16
6.2 PROGRAM MODULES (PM).....	17
6.2.1 PID Modules (PID).....	17
6.2.2 Calculators.....	18
6.2.3 Comparator.....	19
6.2.4 Programmable Logic Control (PLC).....	19
6.2.5 Totalization Module.....	19
6.3 ANALOG OUTPUTS (AO)	20
6.4 DIGITAL OUTPUTS (DO)	20
6.5 DIGITAL CONSTANTS (DCO).....	20
6.6 LOGIC RESULT STATUS' (LRS)	20
7. CONTROLLER COMMISSIONING: STRATEGIES AND PERFORMANCE.....	20
7.1 THE COMMISSIONING TOOL	20
7.2 CONTROL STRATEGIES AND PERFORMANCE	22
7.2.1 Suction Pressure Control.....	22
7.2.2 Discharge Pressure Control.....	23
7.2.3 Mixed Temperature Control.....	25
7.2.4 Suction Temperature Control.....	25
7.2.5 Final Control Strategy.....	27
7.2.6 Final Control Flow Diagram	30
8. AUTOMATIC TEST SEQUENCE SYSTEM.....	32
9. CONCLUSIONS AND RECOMMENDATIONS.....	33
LIST OF REFERENCES	36
APPENDIX A: SYSTEM DIAGRAMS.....	37

APPENDIX B: HP VEE PROGRAM CODE..... 40

APPENDIX C: CONTROLLER CONFIGURATION..... 45

APPENDIX D: VISSIM PROGRAM DESCRIPTION..... 70

APPENDIX E: OPERATING INSTRUCTIONS 101

 E.1 HARDWARE SETUP 102

 E.2 SOFTWARE SETUP 103

 E.3 CHECKLISTS..... 105

E.3.1 Pre-Operation Checklist..... 106

E.3.2 Operation Checklist 106

E.3.3 Post-Operation Checklist 107

LIST OF FIGURES

Figure 1: Basic System Schematic	6
Figure 1: Original Ideal Cycle p-h Diagram	7
Figure 2: Modified Cycle p-h Diagram	8
Figure 3: Load Stand Front Panel	10
Figure 4: Original Control Strategy Feedback Loops	21
Figure 6: Suction Pressure Controls	22
Figure 7: Condenser Flow Rate Effect	23
Figure 8: Discharge Valve with All Cold Condenser Water	23
Figure 9: Preliminary Suction Temperature Controls	24
Figure 10: Final PI Loop Suction Temperature Controls	25
Figure 11: Integral Control of Suction Temperature	26
Figure 12: Comparing PI to Integral Suction Temperature Controls	27
Figure 13: Overall Final Control Strategy Performance	28
Figure 14: Controls Flow Diagram	30
Figure 15: Power Input Measured Data	34
Figure 16: Mass Flow Measured Data	35
Figure 17: Discharge Temperature Measured Data	35

1. Introduction

The purpose of the load stand utilized in this project is to accurately measure the operating characteristics of hermetic compressors, for a range of cooling capacities from 1 ton to 3 tons of refrigeration. The results will be used for comparison to results obtained by a mathematical model developed by Halms[1]. In this project a 1 ton, horizontal type, scroll compressor was tested with R-22 as the working fluid. The purpose of this research project was to commission this load stand, which includes setting up the hardware, setting up a control system, a data acquisition system, and an automatic test sequence system. The objective of the control system is to obtain test points that are defined by a compressor suction pressure, suction temperature, and discharge pressure. The data acquisition system should accurately measure the operating points of the compressor to include power consumption [W], mass flow rate [kg/h], and discharge temperature [$^{\circ}$ C]. These results can then be used to produce a compressor map, verify existing compressor maps, or verify the results obtained from a compressor model. The purpose of the automatic test sequence system is to provide a system that will run the load stand through test conditions without the need for human interactions. This report is not only a research report but also serves as a user's manual for the load stand. It will provide the user with a working knowledge of the load stand and documentation for the software used to operate, control, and modify the load stand systems. The report will include descriptions of the cycle utilized, the system hardware, the data acquisition system, the control strategy and hardware utilized, the operating characteristics of the system, and the automatic test sequence system. It will also provide information useful for changing the system when needed and to run the system effectively. In addition, test results of the 1 ton scroll compressor will be provided and lastly, conclusions and recommendations for system improvement will be given.

2. CYCLE DESCRIPTION

The cycle utilized for the compressor load stand is a hot gas bypass cycle which features some of the characteristics of a vapor compression cycle often found in heating, ventilating, air conditioning, and refrigeration (HVAC&R) systems. In order to obtain the desired operating conditions including suction pressure, suction temperature, and discharge pressure the basic cycle was modified to provide appropriate control devices. As the basic system schematic in Figure 1

shows, the original system hardware consisted of a compressor, a water-cooled condenser, an electrically actuated globe valve to control the condenser water flow, two electronic expansion valves (EEV's), a mixing device, electric heat tape controlled by an SCR (Silicon-Controlled-Rectifier), an oil separator (OS), a coriolis effect flow meter (FM), and a solenoid valve (SV). The numbers in circles correspond to the state points shown in the ideal cycle pressure-enthalpy (p-h) diagrams in Figures 2 and 3. The original system was modified to provide better discharge pressure control by installing the manual control valve located on the discharge side of the compressor. A more detailed discussion of the discharge pressure control problem will be presented later in Section 7.2.2 Discharge Pressure Control. An electrically actuated valve was installed later to provide full automatic controls of the load stand. Full descriptions of the system components can be found in Halms[1].

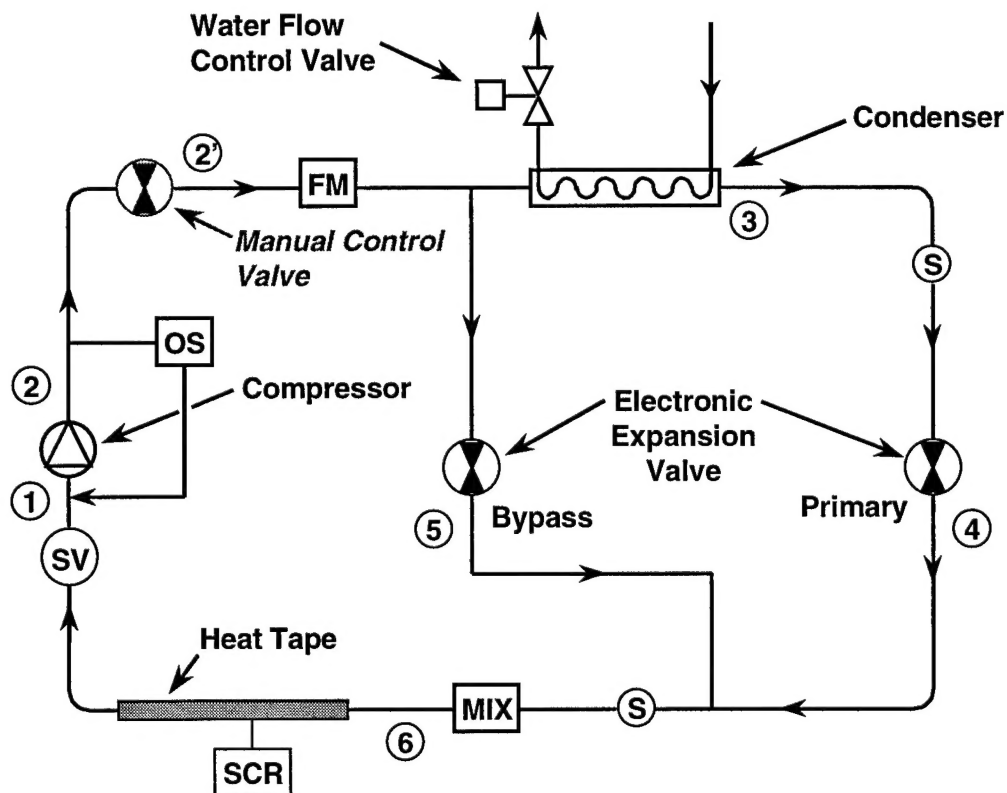


Figure 5: Basic System Schematic

The original ideal cycle is shown on a p-h diagram in Figure 2. To step through the cycle, start at the discharge of the compressor (point 2). The oil is separated from the refrigerant using an oil separator. This is done to get as much oil out of the refrigerant before passing the

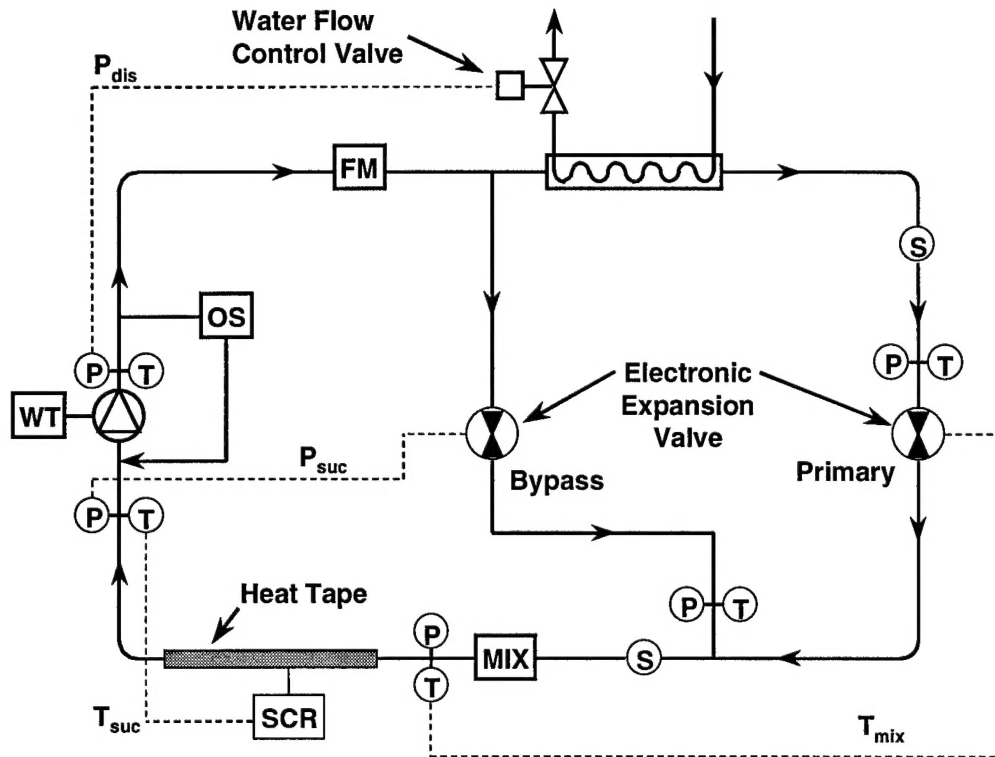


Figure 6: Original Ideal Cycle p-h Diagram

refrigerant through the flow meter so only the mass flow of refrigerant is measured. After passing through the oil separator the refrigerant passes through the flow meter.

From there the refrigerant flow is split and either goes into the primary or bypass loop. The bypass loop consists of an EEV and throttles the fluid to the suction pressure (point 5). The primary loop condenses the refrigerant in the water-cooled condenser (point 3) and throttles it through the primary EEV to the suction pressure (point 4). The two fluids are then mixed (point 6) in a mixing device which consists of several loops of tubing. After mixing, the refrigerant then passes through the electric heat tape region of piping to superheat the refrigerant to the inlet temperature before entering the suction inlet of the compressor (point 1). Figure 3 illustrates the modified ideal cycle p-h diagram. The difference between the original and ideal cycle is that after the discharge of the compressor the refrigerant flow is throttled down to the condensing pressure

using the manual control valve. The rest of the cycle is the same as the original cycle. A more detailed piping schematic of the system is provided in Appendix A.

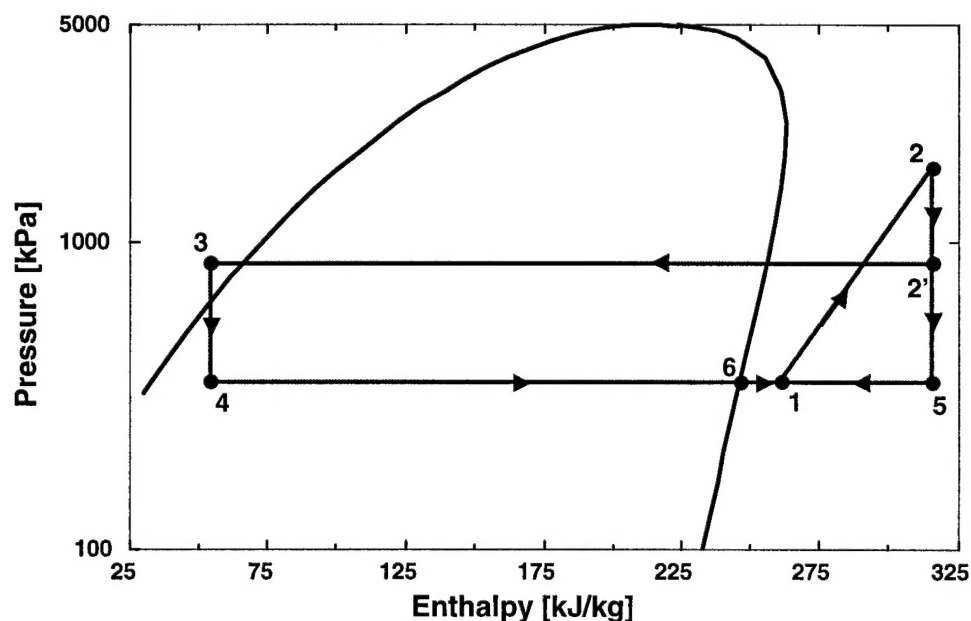


Figure 7: Modified Cycle p-h Diagram

3. HARDWARE SETUP AND DESCRIPTION

As stated previously, the load stand was configured to test various compressors ranging from 1 ton to 3 tons of refrigeration. The design operating conditions are:

Evaporating Temperatures: -25°C to 15°C

Condensing Temperature: 25°C to 65°C

Superheat: 10°C

Subcooling: 10°C

Refrigerant: R-22

The system was sized to accommodate the largest compressor (3 tons of refrigeration). Because the system needs to accommodate different capacity compressors some of the system components are oversized for smaller compressors like the 1 ton scroll compressor that was tested as part of this project. Because some of the components and piping are oversized for the 1 ton scroll

compressor, the original control strategy was unacceptable and had to be modified. The control problems will be discussed in Section 7.2 Control Strategies and Performance.

3.1 Measuring Devices

As the isometric piping diagram in Appendix A illustrates, there are multiple pressure transducers, Type-K thermocouples, and RTD's located throughout the system. The system also includes a mass flow meter located on the discharge line of the compressor and a watt transducer to measure the power supply to the compressor. The RTD's are used solely for control aspects while the thermocouples are used for data acquisition because the Johnson Controls controller will only accept a passive signal from RTD's. A further description of the controller will be provided in Section 5 Controller Description. The thermocouples used for the suction and discharge temperature of the compressor are dwell thermocouples, while the rest of the thermocouples are soldered on the outer surface of the copper tubing. The dwell thermocouples were used at these points because they are the points that are of utmost importance. The other thermocouples are used only for system analysis. If more thermocouples are added to the system and another data acquisition card is required, the user must ensure that the new card has a thermocouple reference device. Platinum, 1000 Ω RTD's were used due to their temperature-voltage linearity. Because the controller uses the RTD's which are mounted on the surface of the tubing, there is a temperature difference between the fluid and what the controller actually "sees". To compensate for this the controller set points must be set above the desired set point. As the fluid temperature gets further away from the ambient temperature the difference increases. Therefore the greatest difference observed is the discharge pressure where differences of up to 20°C were experienced. The discharge temperature seen by the controller is not critical. The controller only uses the discharge temperature for the high temperature cut-off switch and for the temperature display on the front panel. To compensate for the temperature difference in the discharge pressure the high temperature cut-off should be set approximately 10°C below the actual cut-off temperature. For the suction temperature the difference in the fluid and tubing temperature was much less; usually less than 2°C. The pressure transducers are used for controls and for data acquisition. The system has 0-500 psi transducers on the low side and 0-1000 psi transducers on the high side. Further explanation of the data acquisition system is discussed in Section 4 Data Acquisition System.

3.2 Electrical System

A detailed wiring diagram of the system is included in Appendix A. The system requires many different voltages: 230V/3 ϕ /60Hz, 230V/1 ϕ /60Hz, 110 VAC, 24 VAC, and 12 VDC. The load stand uses the 230V/3 ϕ /60Hz supply power and transforms it into the required voltages using multiple transformers. As new compressors are tested the voltage to the compressor may need to be changed to accommodate the specific compressor power input. Also, as larger compressors are tested the fuses will need to be checked for capacity. The current fuses are rated at 5 amps for the 1 ton scroll compressor. The current 1 ton scroll compressor uses 230V/3 ϕ /60Hz power.

3.3 Load Stand Cabinet Description

A photograph of the front panel of the load stand cabinet is shown in Figure 4. The system has two power switches located on the load stand front panel; one switch for the system main power ("SYSTEM" switch) and another for the compressor power ("COMP" switch). The lights on the cabinet indicate that the various voltage circuits are energized. The three inline green, red, and amber lights indicate 12 VDC, 24 VAC, and 115 VAC respectively. The separate red light indicates 230V/3 ϕ /60Hz power. The four displays mounted on the cabinet show suction and discharge pressures (bars) and temperatures ($^{\circ}$ C). As the picture shows, the discharge

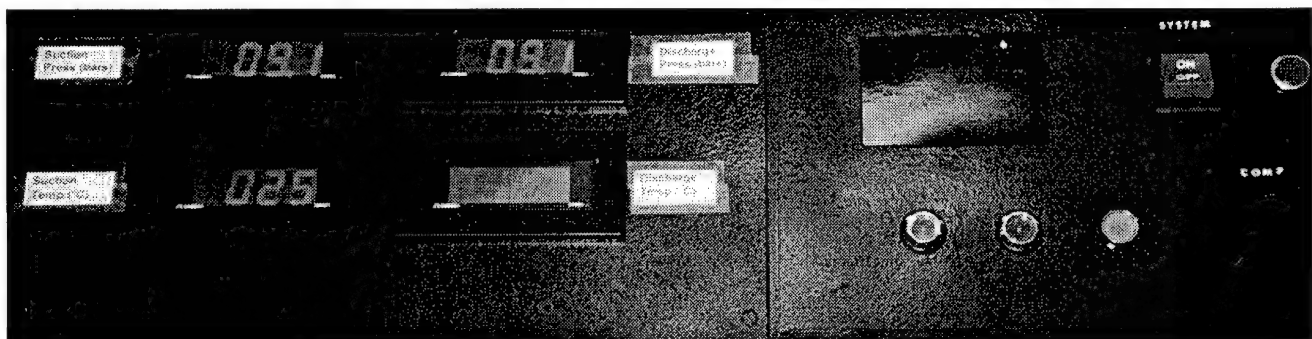


Figure 8: Load Stand Front Panel

temperature display is currently not functional and needs to be replaced. The blanked-out square hole above the lights in the panel is to accommodate the inverter control panel when the required cable is received. The pressure displays mounted on the front panel of the cabinet are wired

directly to the pressure transducers. The temperature displays are connected to analog outputs from the controller because they require a voltage and the RTD's do not output voltages. The controller is used to convert the signal from the RTD's to a voltage used by the displays.

3.4 Condenser

The system uses tap water for the condensing fluid with two hose bib connections (supply and return) located on the right rear of the unit with appropriate shut-off valves located on the right side of the unit. As the schematic in Appendix A illustrates, the system utilizes an electronically actuated globe valve on the condenser water return for condenser water flow control (condenser pressure/temperature control). This valve is controlled directly by the controller using a 0-10VDC signal. Section 7.2.2 Discharge Pressure Control will discuss the problems encountered with condenser pressure and temperature control.

3.5 Electronic Expansion Valves

Electronic expansion valves are used as the expansion devices. Circuit boards that are mounted vertically next to the controller control the valves. The circuit boards receive either a 0-10VDC or 0-20mA signal from the controller to adjust the valves. The boards have a jumper to set the appropriate input signal. Currently the primary and bypass EEV's receive 0-10VDC signals and the new discharge pressure control EEV receives a 0-20mA signal. Additional guidance on how to set the analog output signals of the controller will be provided in Section 6.3 Analog Outputs (AO). Caution should be taken if additional boards/valves are installed. The power supply for the boards must be isolated from the power supply for the controller. If this is not done, the board will be damaged and will need to be replaced.

3.6 Compressor

The current scroll compressor utilizes an inverter to provide power to the compressor. Adjusting the frequency in the inverter can change the speed of the compressor. This is accomplished by opening the cabinet lid and pushing the mode button on the inverter (located in the front right of the cabinet) and pressing the arrow buttons to adjust the frequency to the desired speed (Hz). Once the desired frequency is adjusted, push the "RUN" button to start the compressor. To stop the compressor, push the "STOP" button on the inverter. Once the inverter control panel is moved to the load stand front panel, the user will be able to make all adjustments to the inverter without opening the lid to the load stand cabinet. A safety switch is installed on

the lid of the cabinet. The system will shut down if the lid is opened for safety reasons. The switch is currently bypassed to allow the user to open the cabinet lid and make adjustments to the compressor inverter.

3.7 Safety Devices

To protect the compressor, a mechanical, high-pressure cut-off switch was installed. The high-pressure cut-off is currently set at approximately 375 psi (2586 kPa). A blow-off valve was also installed in the discharge line to protect the compressor. The blow-off valve is set at 400 psi (2758 kPa). Additionally, the controller is programmed to open the compressor relay (turn off the compressor) if a high discharge pressure of 2500 kPa (363 psi), high discharge temperature of 110°C, or low suction pressure of 100 kPa (14.5 psi) is encountered. The controller will close the relay and reenergize the compressor when the pressures/temperatures are back into the "safe" region which are currently defined as a discharge pressure less than 1500 kPa (218 psi), suction pressure greater than 600 kPa (87 psi), and discharge temperature less than 80°C. Because the inverter sustains a charge for approximately 20 seconds after power is disconnected the compressor will start running again if the relay is closed before the inverter charge is depleted. When the inverter charge is depleted it will not turn back on, even if the relay is closed, until manually reset by pressing the "RUN" button on the inverter.

3.8 Refrigerant Piping Accessories

The system uses an oil separator as previously discussed in Section 2 Cycle Description to remove as much oil from the refrigerant flow before passing it through the flow meter. An accumulator is installed on the suction of the compressor to hold the surplus refrigerant. A filter/drier is installed in the discharge of the condenser. Sight glasses at the condenser outlet and the mixing point after the EEV's are located on the front of the load stand cabinet. There is another sight glass on the side of the cabinet at the outlet of the oil separator to ensure proper oil circulation.

3.9 Solenoid Valve

An electrically actuated solenoid valve is included in the suction line to aid in the start-up and shutdown procedures to prevent liquid from entering the suction of the compressor. The original intent was to isolate liquid refrigerant from the suction of the compressor at shutdown. When the start-up procedure is initiated the electric heat tape will heat the refrigerant on the

suction side of the compressor until it is in a superheated state. At that point the solenoid valve will be opened and the compressor started. This valve has not been necessary yet because R-22 is superheated at room temperature. To enable this type of start-up/shut-down procedure a simple PLC module would provide the appropriate controls.

3.10 Electric Relays

There are three relays connected to digital outputs of the controller. These are required to enable the controller to turn on/off the various devices. One relay is for the SCR. The SCR is a control device that receives a 4-20 mA signal from the controller and varies the electric input to the electric heat tape (0-936 kW). The controller also has digital outputs for the solenoid valve relay and compressor relay. Information on the various inputs/outputs to the controller will be discussed in Section 7 Controller Commissioning: Strategies and Performance.

4. DATA ACQUISITION SYSTEM

The load stand uses a Hewlett Packard HP 75000 Series B unit to receive the input signals from the various data acquisition devices. The load stand cabinet has quick disconnects located on the left side for easy connection to the data acquisition system from the various sensors located throughout the system. The unit accepts either voltages or thermocouple inputs (has a reference point to convert mV to temperature). The amperage from the mass flow meter is converted to a voltage signal through the use of a 470 Ω resistor, which converts the 4 to 20mA signal to a 1.88 to 9.40 VDC signal. The computer is able to read the various inputs to the HP 75000 through the use of the software package HP Vee. The HP Vee program written for the load stand, *loadstnd.vee* (printout included in Appendix B), converts all the input signals to the appropriate units (kPa, °C, and kg/hr) and displays them on the screen in a graphical and numeric format to aid in system monitoring. The system also has the ability to save the data collected in a tab delimited text file that can be imported into a spreadsheet for later analysis. To write data to a file the user should check the "Write Data" box on the display panel. The frequency at which data points are collected and displayed or written to the data file can be adjusted in the program. The user should change the "Write Freq" and "Disp Freq" values to the amount of time (seconds) desired between data points collected. As the data points are collected the plots and numeric displays will be updated on the screen.

To get into the details of the program, the program starts at the start button. The lines indicate the path in which the program executes the various functions. The cycle timers are used to trigger the functions at the specified time intervals. The first function processed by the program is the write to file checkbox. If the "Write Data" checkbox on the display panel is checked the program prompts the user for a filename to write the data to and a summary of the experiment being conducted. After the user inputs the required information, the data file is created with the current date and time and summary information provided by the user. The program execution continues by starting the different cycle timers. If the "Write Data" checkbox is not checked the program starts automatically. Note that the "Write Data" checkbox cannot be checked while data is being taken. To start writing data during an experiment click the "STOP" button, check the "Write Data" box, push the "Start" button, and provide the required information. The data file puts all the data points collected into columns with appropriate headers and the elapsed time at each data collection point.

There are two cycle timers in the program. The "Write Freq" variable set by the user in the display panel determines how often data points are written to the data file. Similarly, the "Disp Freq" variable set by the user determines how often the display values are updated. Note that the display frequency should be less than or equal to the write frequency because the variables used for writing data are collected in the display loop. The display loop reads the input signals and exports these values to formula blocks that convert the values into the appropriate units. The value is then set to the appropriate variable. The variables are then used to write the data to the data file and display the value on the display panel.

The data acquisition system is very sensitive to other equipment in the lab that might be running and causing interference in the electrical power supply system. During initial testing, when the psychometric rooms were running the data collected was not accurate and many error signals were received by the system. When the psychometric rooms were shut down a dramatic change in the data collection was observed (drop in temperature and pressure) and the readings returned to accurate data points. The program written for the system filters out many of the error signals when writing to the data file and logs these error signals in a field labeled "error codes". If an error signal is received, the previous "good" signal is kept as the current point and an entry is added to the error code identifying which error signal was observed. Throughout initial testing

the voltage signals from the pressure transducers (especially the suction and discharge) were not very clear and error signals were received sporadically. Many times the suction and discharge pressures were not read at all. This was eliminated by installing jumpers between the low signal from the transducer and the ground on the data acquisition card. Table 1 lists the Channels used, their corresponding data collection device, and the corresponding input signal and range.

Table 1: Data Acquisition Channels

Channel	Data Point	Label	Input Signal	Signal Range
0	Suction Temperature	T_1	mV	Type K Thermocouple
1	Discharge Temperature	T_2	mV	Type K Thermocouple
2	Condenser Outlet Temperature	T_3	mV	Type K Thermocouple
3	Primary EEV Outlet Temperature	T_4	mV	Type K Thermocouple
4	Mixed Temperature	T_5	mV	Type K Thermocouple
6	Suction Pressure	P_1	1-6 VDC	0-500 psi
7	Discharge Pressure	P_2	1-6 VDC	0-1000 psi
8	Condenser Outlet Pressure	P_3	1-6 VDC	0-1000 psi
9	EEV's Outlet Pressure	P_4	1-6 VDC	0-500 psi
10	Mixed Pressure	P_5	1-6 VDC	0-500 psi
11	Mass Flow Rate	m_dot	1.88-9.4 VDC	0-125 lbm/hr
12	Power Input	N/A	0-9.4 VDC	0-3 kW

Channel 12 is wired for the watt transducer but no data could be collected from the watt transducer yet. Data from the watt transducer was collected manually using a voltmeter. Channel 13 is pre-wired into the card for any additional device that might be installed in the system. Both channel 12 and 13 are not programmed into the HP Vee program developed. The card has a total of 16 channels. There are currently four channels that are unused (5 and 13-15). Channel 5 was left blank to facilitate the addition of another thermocouple. The HP Vee program scans sequential channels using a multi-device function. By keeping all similar devices in sequential channels it facilitates easier programming in HP Vee.

5. CONTROLLER DESCRIPTION

The controller in the load stand is a Johnson Controls DX-9100, Version 2. The controller accepts 8 analog inputs (AI) each of which may be 0-10 V, 0/4-20 mA, or a passive RTD sensor. Jumpers on the bottom of the controller must be installed for all input signals. In addition jumpers can be installed for "Fail High" or "Fail Low" voltage inputs. For 0-20 mA signals, a zero offset to 4 mA can be set using the software configuration. The passive RTD inputs are configured to convert the measured resistance to the appropriate units (either °C or °F configured by software). The controller provides 15 VDC to the active inputs. The DX-9100 can also accept 8 digital inputs (DI) but none were used on the load stand. The DX-9100 provides eight analog outputs (AO) of which four can be set to 0-10 V or 0-20 mA (4-20 mA by software configuration) and four outputs that are 0-10 V only (AO11-14). Jumpers must be installed for AO1-2, AO9-10 for the type of output signal desired. The DX-9100 also has six digital outputs (DO), eight analog constants (ACO), 32 digital constants (DCO), and 64 Logic Result Status (LRS) variables. The controller provides real time functions, 12 programmable function modules, and one programmable logic control (PLC) module. Switches on the bottom of the controller must be set for the controller address on the N2 bus, which is currently set to 1.

6. CONTROLLER PROGRAMMING

The controller is programmed through the use of Johnson Control's GX-9100 Programming Tool Version 3.0. The program communicates with the controller using the N2 bus connected to the COM port of the computer through a Metasys adapter. It can also communicate through the use of an RS-232 cable. Using the software the user can program the inputs, control modules, constants, PLC's, and outputs. The following information on programming the controller has been adapted using Johnson Controls[2].

6.1 Analog Inputs (AI)

By programming the analog input module the user can configure the controller for the types of inputs used. To configure an active input (other than RTD's), change the type of active input to:

0: 0-10 VDC

1: 4-20 mA

2: 0-20 mA

For passive inputs (RTD's) set the type of passive input to:

1: Ni1000 (JCI characteristic)

2: Ni1000 extended temperature range

3: A99 (JCI characteristic)

4: Pt1000 (DIN characteristic)

5: Ni1000 L & G

6: Ni1000 (DIN characteristic)

For the current RTD's, the passive input type is set to 4. For active inputs enter the High Range (HR) and Low Range (LR) corresponding to the sensor range and output range according to the following equation: $AI = (\text{input}/100) * (HR - LR) + LR$. Where input is the analog input signal in % of physical input signal. For example, the suction pressure transducer outputs a 1-6 VDC signal for 0-500 psi. To set the analog input (AI1) for conversion to kPa, the High Range (10 VDC) is:

$$HR = (500 \text{ psi} + \frac{500 \text{ psi}}{5 \text{ VDC}} * 4 \text{ VDC}) * 6.895 \frac{\text{kPa}}{\text{psi}} = 6205.3 \text{ kPa}$$

Similarly, the Low Range (0 VDC) is:

$$LR = (0 \text{ psi} - \frac{500 \text{ psi}}{5 \text{ VDC}} * 1 \text{ VDC}) * 6.895 \frac{\text{kPa}}{\text{psi}} = -689.5 \text{ kPa}$$

For passive inputs (RTD's) the ranges are set automatically according to the sensor type.

6.2 Program Modules (PM)

This report will only cover Proportional-Integral-Derivative (PID) modules, calculators, comparators, totalization modules, and Programmable Logic Control (PLC) modules because they were the only PM's used for the load stand.

6.2.1 PID Modules (PID)

The programmer must set various parameters in the PID modules. The parameters that are essential to proper operation of the module are:

Process Variable (PV): connect the analog input signal (AI) to the PV connection on the PID module. This determines what input the module will be controlling. Note that the signal will be in the appropriate units defined in the analog input module and not a voltage or amperage.

Local Set Point (LSP): set point that the module is trying to achieve in appropriate units.

Proportional Band (PB): percent of the input signal that will produce a full-scale change in the output control signal (OCM). For example, for an analog input range of 0-100, a PB of 10 dictates that a change in the PV of 10 units will cause a 100% change in the OCM. A negative value for the PB will cause a reverse acting module; i.e. if the PV increases the OCM will decrease.

Reset Action (TI): defines the integration time and is expressed in repeats per minute. The integral time is determined by $1/TI$. Thus, a larger value of TI increases the integral gain.

Rate Action (TD): defines the derivative action decay time parameter. Derivative action was not utilized in any load stand control strategy and therefore will not be discussed. To disable the derivative make sure the TD is set to zero.

Error Deadband (EDB): expressed as a percent of the PB. If the PV is within the EDB the module goes to proportional only control. To disable it set the EDB to zero.

Output High/Low Limit (HIL/LOL): expressed as a percent of the output. Defines a high or low limit that the OCM will not go above or below.

Deviation High High/Low Low Limit (DHH/DLL): expressed as a difference between the PV and the LSP. When the PV is above/below these limits the controller goes to proportional only control if the PID to P parameter is set to one. This is used to prevent anti-windup of the integral action.

6.2.2 Calculators

The only calculator used in any load stand control strategy is the eight-channel calculator. The module can take up to eight input variables and perform an algebraic expression on those inputs according to the equation chosen (two equations provided). A high and low limit for the output can be set also.

6.2.3 Comparator

The comparator module is used to compare an input signal to a set point. It compares the input signal (In) to the set point (SP) and differential (DF). It then sets a Logic Status (LS) variable depending upon the type of comparator chosen. The differential is the amount above/below the set point which will cause the comparator to reset the LS. The different types of comparators are:

- | | |
|----------------------|--------------------------------------------------------------------------------------|
| 1 = High Limit: | LS = 1 when $In \geq SP$
LS = 0 when $In \leq SP - DF$ |
| 2 = Low Limit: | LS = 1 when $In \leq SP$
LS = 0 when $In \geq SP + DF$ |
| 3 = Equality Status: | LS = 1 when $SP - DF < In < SP + DF$
LS = 0 when $In < SP - DF$ or $In > SP + DF$ |

6.2.4 Programmable Logic Control (PLC)

A PLC is used on the load stand for the high/low safety switches. A PLC is also used to reset the totalization module integral. The PLC consists of sequential user-defined instruction blocks used to set logic states. The first instruction of a PLC block must begin with a LOAD or LOAD NOT instruction (like an IF or IF NOT). Subsequent instructions are carried out using AND/AND NOT/OR/OR NOT blocks and at the end of the instruction block an OUT or OUT NOT instruction to set the logic state.

6.2.5 Totalization Module

A totalization module was used in the final load stand control strategy. The totalization module can be configured in a couple different ways but the load stand control strategy uses it to perform a numeric integration on the difference between the suction temperature and the set point. The user must input the set point either manually or through the use of an input from another control module. The time constant must also be set. The time constant determines the time between numeric integrations. A smaller time constant will increase the integration frequency but will also provide a more accurate result.

6.3 Analog Outputs (AO)

AO's are used to control the various control devices installed in the system such as the EEV's. The module receives a control signal and converts that into a voltage or amperage to adjust the device. The parameters that need to be programmed by the user are:

Source Point (AO@): this input defines where the module gets the control signal. To connect a PM to an output, connect the OCM connection from the PM to the AO@ connection on the output. The OCM is expressed as a percent of the analog output range.

High/Low Range (HRO/LRO): defines the OCM that will cause a 100% or 0% output signal.

6.4 Digital Outputs (DO)

Only On/Off digital outputs were utilized in any load stand control strategy. The output is either on or off depending upon the input, which can be a logic source or a numeric value. A positive numeric value turns the output on and a zero or negative numeric value turns the output off. These outputs are used to turn on and off the various relays on the system.

6.5 Digital Constants (DCO)

Digital constants can be set to either one or zero. They are used as inputs to the digital outputs, a PLC, or a PM.

6.6 Logic Result Status' (LRS)

An LRS is either set by an OUT, OUTNOT, SET, or RST instruction in the PLC module. All load stand strategies only use the OUT or OUTNOT instructions. They are used on the load stand to control the digital outputs.

7. CONTROLLER COMMISSIONING: STRATEGIES AND PERFORMANCE

7.1 The Commissioning Tool

A useful tool in testing control strategies and analyzing the effect of changing parameters in the controller is the Johnson Controls Point Template software package. This software allows the user to access the controller settings "on the fly" (change them manually without having to

The diagram illustrates a water flow control system for a heat exchanger. Key components and their connections are as follows:

- Water Flow Control Valve:** Located at the top, it regulates the flow of water into the system.
- FM (Flow Meter):** Measures the flow rate of water entering the heat exchanger.
- OS (Orifice):** A restriction in the flow path used for flow measurement.
- WT (Water Tank):** The source of water for the system.
- P (Pressure) and T (Temperature) Sensors:** Multiple sensors are placed throughout the system to monitor flow and temperature. Specifically, there are pressure sensors at the inlet and outlet of the heat exchanger, and temperature sensors at the inlet and outlet of the heat exchanger, as well as at the bypass and primary lines.
- SCR (Solenoid Control Relay):** Controls the flow of water through the heat exchanger.
- Heat Tape:** A heating element used to maintain the temperature of the water in the heat exchanger.
- MIX (Mixer):** A device that mixes the water from the bypass and primary lines.
- Bypass:** A line that allows water to bypass the heat exchanger.
- Electronic Expansion Valve:** A valve that controls the flow of water through the primary line.
- Primary:** The main line of the system, which carries the water from the heat exchanger to the bypass and primary lines.
- S (Sensor):** A sensor located at the outlet of the primary line.

The flow path starts from the water tank, passes through the flow meter, orifice, and heat exchanger. The water then splits into a bypass line and a primary line. The bypass line goes through a mixer and back to the heat exchanger. The primary line goes through an electronic expansion valve and back to the heat exchanger. The system is controlled by a water flow control valve and a solenoid control relay.

Page 21

control loops to see the effect of multiple loops running simultaneously. The user will have to adjust the response of the loops to develop a control strategy that will work effectively.

7.2 Control Strategies and Performance

The original control strategy developed for the load stand is shown schematically in Figure 5. The dashed lines indicate the different feedback control loops utilized. The general procedure used for tuning the PI modules is as follows:

1. Set the PB for a value that should give an appropriate change in OCM for the change in the PV defined by the PB. Increasing the PB will cause the OCM to change less for the same change in PV. For example, on the bypass valve the PB is set at 6% which translates into a 100% change in the OCM for a PV change of $0.06(6894 \text{ kPa}) = 414 \text{ kPa}$.
2. Set the Reset Action (TI) to a small value initially.
3. Try the current settings. If there are oscillations or the loop is unstable, either increase the PB or decrease the TI (integral gain). If the response is too slow the opposite action of an oscillatory response should be taken.

7.2.1 Suction Pressure Control

As Figure 5 shows, the Bypass EEV controls the suction pressure. A reverse acting PI module was used for this loop. The suction pressure is a quick responding variable and because

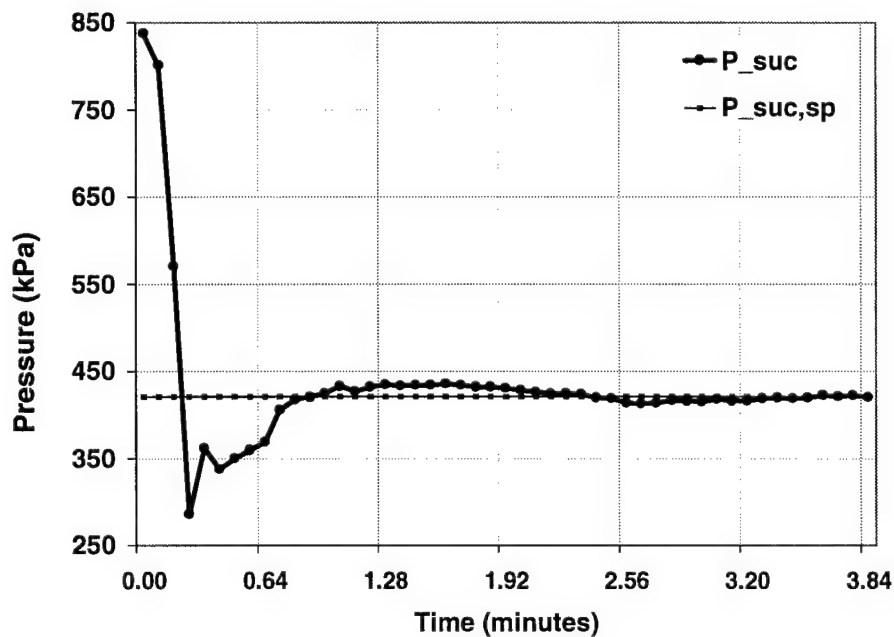


Figure 6: Suction Pressure Controls

of this the integral gain can be set higher than in the other control modules. Figure 6 shows the resulting response, which is very effective in achieving steady state. A full printout of all final controller settings is included in Appendix C.

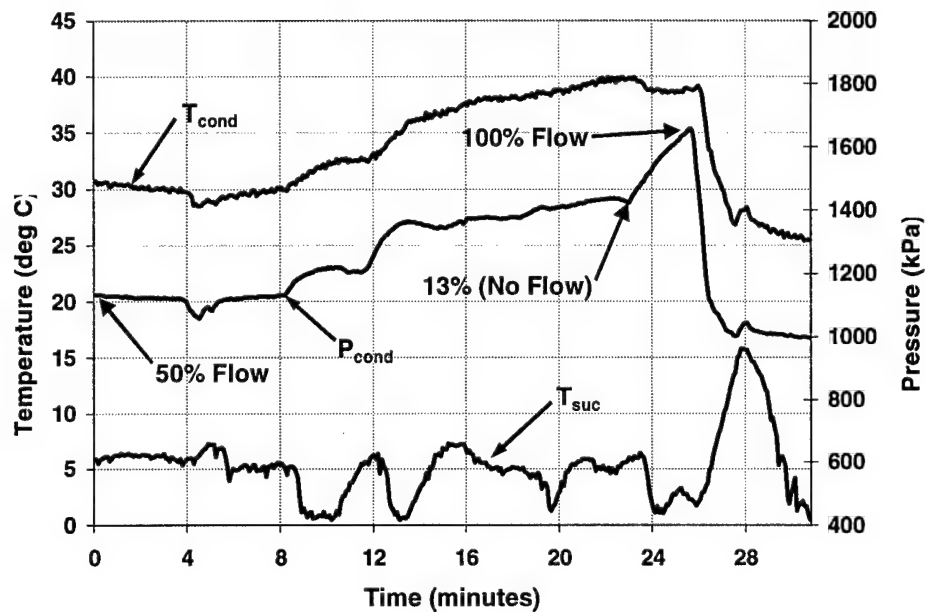


Figure 7: Condenser Flow Rate Effect

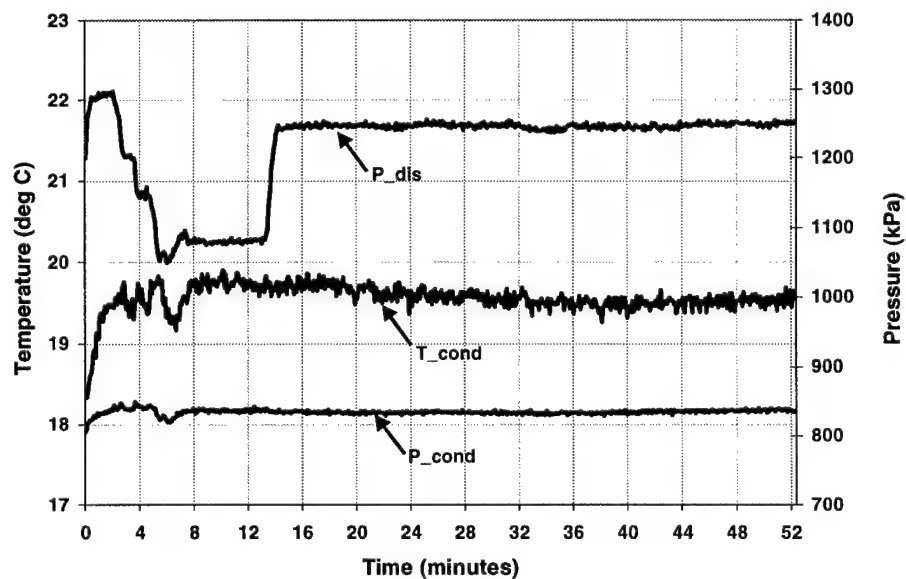


Figure 8: Discharge Valve with All Cold Condenser Water

7.2.2 Discharge Pressure Control

The discharge pressure, which is the same as the condensing pressure, was originally set up to be controlled by the condenser water flow valve. By varying the flow rate of condenser

water the condensing or discharge pressure can be adjusted. The PI loop used for this was a direct acting loop. After testing the control strategy it became apparent that the water flow rate was ineffective in controlling the discharge pressure. The discharge pressure could not be increased enough to achieve the large pressure differences required. This may be due to the large capacity of the condenser. When larger capacity compressors are tested in the future this strategy might be effective in controlling the condenser pressure.

Originally the water supplied to the condenser was all cold tap water. Because the pressure could not be increased enough, hot tap water was added to increase the condenser temperature and thus the condenser pressure. As Figure 7 illustrates, this was effective in raising the pressure but again the pressure could not be changed enough for the higher pressure differentials. By adding hot tap water to the condenser another problem was introduced. As different sources in the building used hot water the condenser temperature and pressure would change drastically. This prevented the system from achieving steady state. Because of these problems a manual control valve was installed on the discharge of the compressor to control the discharge pressure. By doing this, the condenser flow rate could be fixed and a constant condenser pressure and temperature was achieved. Figure 8 shows the response obtained by using the manual control valve for discharge pressure control. The pressures remain constant

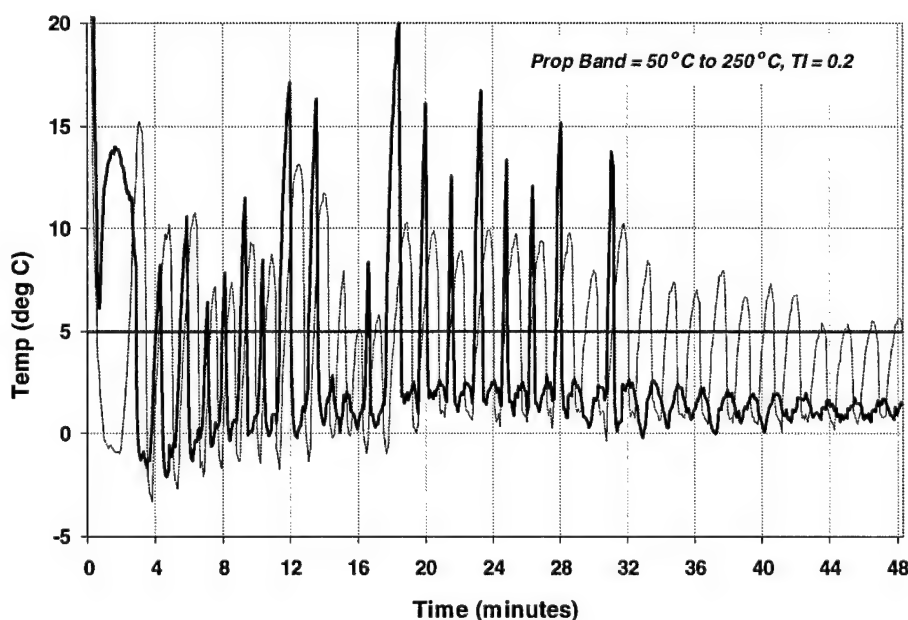


Figure 9: Preliminary Suction Temperature Controls

after achieving steady state rather than oscillating, which happened when hot water was added to the condenser water supply to increase the discharge pressure. Adding the additional control valve was extremely helpful in achieving the required suction state because it provided a constant baseline for the EEV's to operate from.

7.2.3 Mixed Temperature Control

Originally the primary EEV was configured to control the mixed temperature, which is the temperature after the bypass loop and primary loop fluids are mixed. During initial testing the primary valve control loop could not be tuned to provide a stable response.

7.2.4 Suction Temperature Control

As Section 7.2.3 explained, the primary valve was ineffective in controlling the mixed

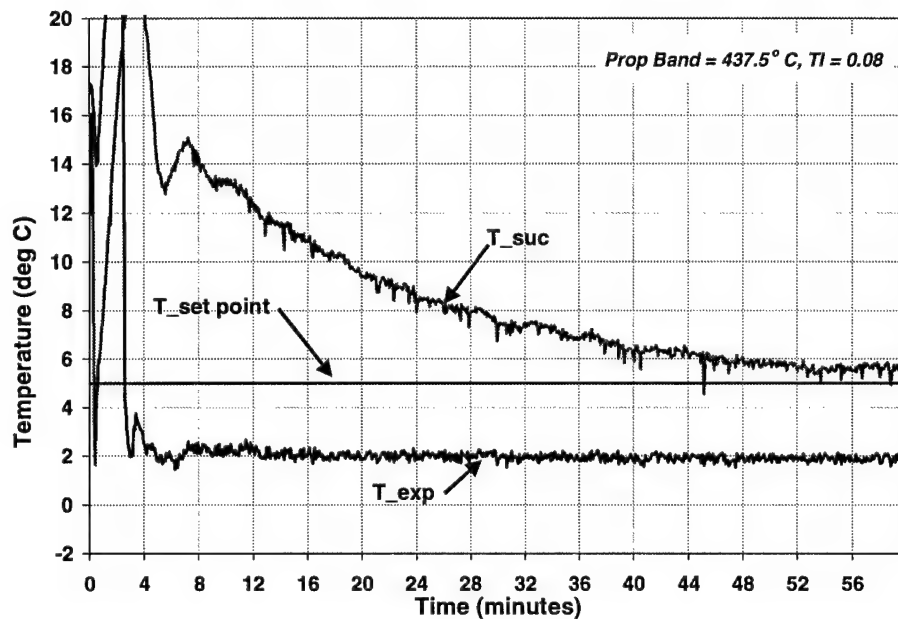


Figure 10: Final PI Loop Suction Temperature Controls

temperature, which caused the suction temperature to remain unstable. Because of this, the primary valve was reconfigured to control the suction temperature. The electric heat tape was also originally configured to control the suction temperature. Figure 9 shows the initial response obtained using this strategy with a small PB and fairly small reset action (TI). As the graph shows the response was not acceptable. The reason for the poor performance is that the bypass and primary valves are fighting each other to reach their set points. As the primary closes to raise the temperature the suction pressure increases also. To counteract the increase in suction pressure,

the bypass opens. This causes more mass flow through the bypass loop which causes the suction temperature to increase even faster. To remedy this situation the primary valve loop must be

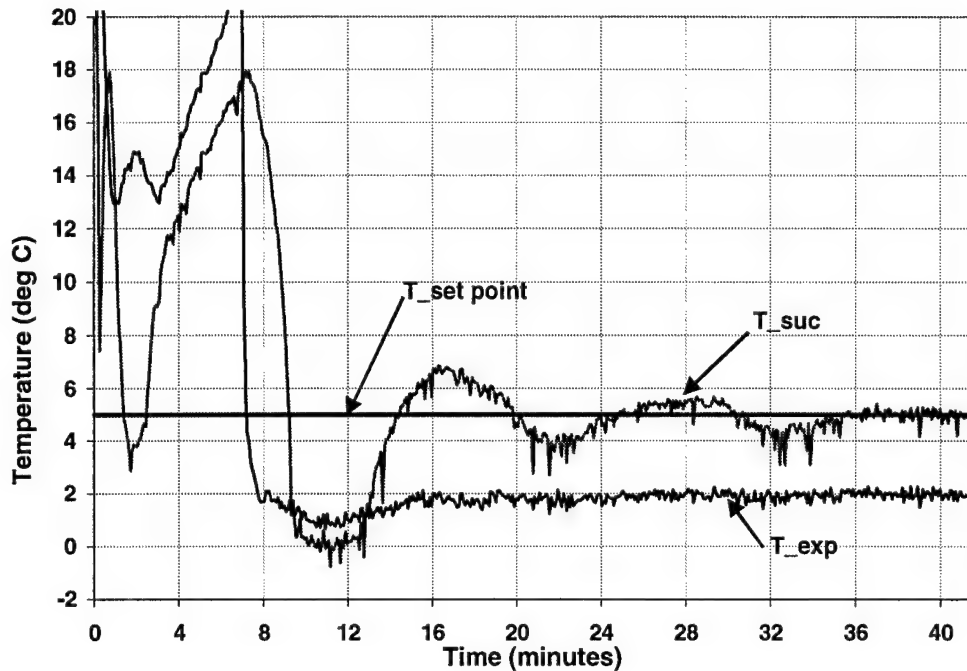


Figure 11: Integral Control of Suction Temperature

slowed down so that the bypass valve can keep the suction pressure constant as the primary valve changes. After slowing down the primary control loop by increasing the PB and decreasing the reset action (integral gain) the response shown in Figure 10 was obtained. The response is stable but it takes a long time to achieve the set point. Due to the slow response of the PI control loop a different strategy was needed.

To remedy this situation a totalization module was used to perform the integration on the suction temperature and the set point. The output is the value of the integral, which is fed into an eight-channel calculator. The calculator takes 10% of the integral signal and uses this as the control signal for the primary valve. A problem with this strategy is that the heat tape is also configured to control the suction temperature so when the PV was below set point, the heat tape would come on. This would counteract the integral action of the primary valve by decreasing the time the PV was below set point. As a result, the primary valve would continue to open and the heat tape would continue to increase its output until the heat tape reached its high limit of 40%.

To remedy this situation the heat tape set point was adjusted to 5°C below the suction temperature set point to avoid the heat tape from augmenting the primary valve. The heat tape only comes on when the suction temperature is close to the saturation temperature. The result is that the primary valve zeroes in on the required valve position and provides a good response Figure 11 illustrates. If we compare the PI controls to the integral controls (Figure 12), it is apparent that the integral control performs better. The following information is very IMPORTANT: At start-up the totalization module is not on so that it will not wind-up during system initialization! Just before the compressor is started the user needs to set the “totalize” digital constant to ON. If this is not done the totalization module will not perform the integration and the primary EEV will remain closed. Setting the “totalize” digital constant to off and then back to on will clear the integral history. This should only be done if the system is shut down.

7.2.5 Final Control Strategy

Appendix C contains a complete listing of the control parameters programmed in the

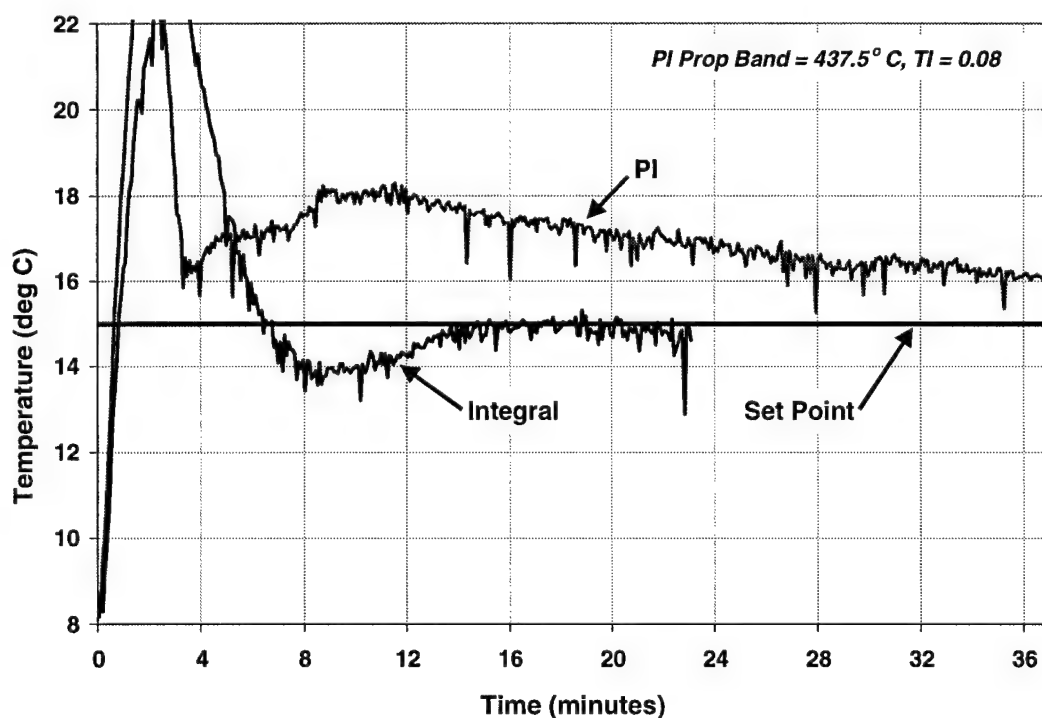


Figure 12: Comparing PI to Integral Suction Temperature Controls

controller. The final control strategy developed includes PI loops for the suction pressure and heat tape control. The heat tape is not necessary for controlling the suction temperature so it was

configured as a safety device to prevent the suction temperature from getting too close to the saturation temperature. The bypass loop is a quick responding PI loop which keeps the suction pressure at steady state very well. The condenser flow control valve is not included in the control strategy. The condenser flow is held constant to provide a constant condenser pressure and temperature. The suction temperature is controlled using an integral action loop for the primary valve. The manual control valve on the discharge of the compressor has been replaced by an EEV and will be controlled by a PI loop.

Currently, there has been no testing of the discharge EEV. It is expected that this valve will provide a quick responding control of the discharge pressure without adversely effecting the

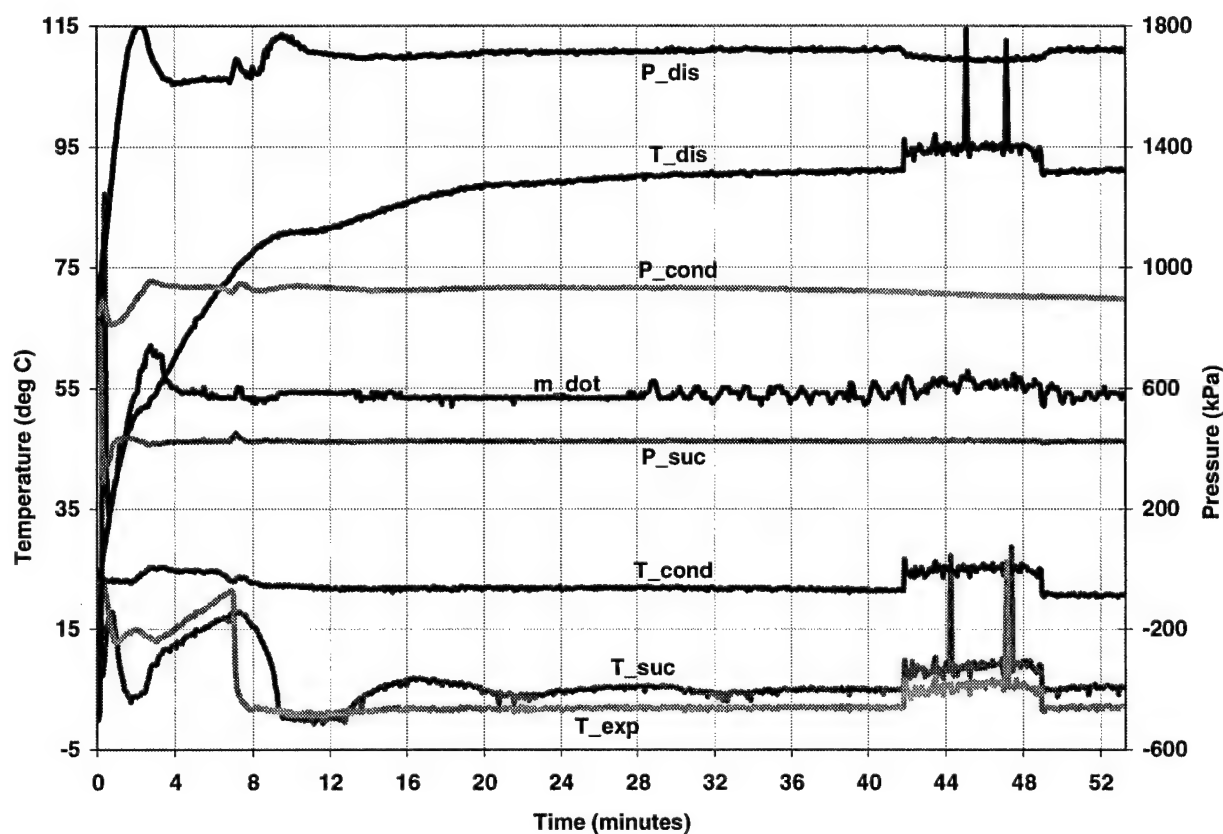


Figure 13: Overall Final Control Strategy Performance

other control loops because the discharge pressure does not effect any other state in the cycle directly. The overall control performance of the final control strategy is shown in Figure 13. The desired test point is a suction temperature of 5°C, suction pressure of 421 kPa, and discharge pressure of 1728 kPa. The response is good because it does not have any unsteady or oscillatory

responses and holds the set points once they are achieved. This experiment was started from an equilibrium state. The next test point should be achieved much quicker. As the plot illustrates, the discharge temperature is the slowest response. Since discharge temperature is not controlled, the control system is effective in achieving the desired set points in a reasonable amount of time.

A comparator control module was utilized to set up the high and low pressure and temperature cut-off switches. The comparator compares the input signal to a set point and depending upon the type of comparison (<, >, =) will either set a logic status to on or off. Using the logic status, a PLC module was created to either turn the compressor relay (LRS 1/DCO 7) and heat tape (LRS 2/DCO 5) on or off. Whenever the compressor is shut off the heat tape is also shut off to prevent the heat tape from being damaged. The "Set point Value #1" is the set point for the high-pressure cut-off in kPa, the "Set point Value #2" is the high temperature cut-off in °C, and the "Set point Value #3" is the low-pressure cut-off in kPa. The differential values are the amount in appropriate units that the input signal has to deviate from the set point before the logic status is returned to zero.

Table 2: Controller Setup

Item	Device	Signal	Range
Analog Input 1 (AI 1)	Suction Pressure Transducer	1 - 6 VDC	0 - 500 psi
Analog Input 2 (AI 2)	Discharge Pressure Transducer	1 - 6 VDC	0 - 1000 psi
Analog Input 3 (AI 3)	Suction Temperature RTD	passive input	(-50) - 250 °C
Analog Input 4 (AI 4)	Mixed Temperature RTD	passive input	(-50) - 250 °C
Analog Input 5 (AI 5)	Discharge Temperature RTD	passive input	(-50) - 250 °C
Analog Output 1 (AO 1)	Bypass EEV	0 - 10 VDC	closed - open
Analog Output 2 (AO 2)	Condenser Flow Control Valve	0 - 10 VDC	closed - open
Analog Output 9 (AO 9)	SCR Controller (heat tape)	4 - 20 mA	0 kW - 1000kW
Analog Output 10 (AO 10)	Primary EEV	0 - 10 VDC	closed - open
Analog Output 11 (AO 11)	Suction Temperature Display	0 - 10 VDC	(-50) - 250 °C
Analog Output 12 (AO 12)	Discharge Temperature Display	0 - 10 VDC	(-50) - 250 °C
Digital Output 3 (DO 3)	Solenoid Valve Relay	0/1	OFF/ON
Digital Output 5 (DO 5)	SCR Relay	0/1	OFF/ON
Digital Output 7 (DO 7)	Compressor Relay	0/1	OFF/ON
Digital Constant 1 (DCO 1)	Shutdown Variable	0/1	OFF/ON

ability to write digital messages to the controller but it is limited to 2 bytes or the integers values -127 to 127. Because of this limitation the set points for the pressures have to be calculated in the controller using calculators as shown in the diagram. The SP1 value is multiplied by 100 and added to the SP2 value. The temperature set points do not have to be calculated because they fall within the 2-byte range. The analog inputs are used as process variables for the program modules and for the comparator. The program modules use the outputs from the set point calculators to set the working set points. The output control signal (OCM) generated by the program modules are then input to the corresponding control device analog output.

The comparator is used to trigger the various high/low pressure and temperature switches (LS's). These LS's are used by the relay control PLC to turn on/off the various relays through the use of LRS's and ultimately the digital outputs. The LRS's are used to set digital constants, which are used as the inputs to the digital outputs. The comparator is also used for the integral control of the suction temperature. The comparator outputs the difference between the suction temperature set point and the actual measured suction temperature. This value is then sent to the totalizer, which keeps a running total of the input signal. The accumulated total is sent to a calculator which determines the primary EEV output value by taking 10% of the accumulated total (integral). The suction and discharge temperature analog inputs are sent to analog outputs 11 and 12 respectively for the load stand front panel displays. This is required as stated previously in Section 3.3 Load Stand Cabinet Description to convert the RTD signals to a voltage used by the displayed. Digital Constant #2, Solenoid, is used to control the solenoid valve. This valve is not used currently as discussed in Section 3.9 Solenoid Valve. Digital Constant #3, Totalize, is used in conjunction with the Reset PLC module to reset the integral in the Totalizer. This would be necessary only if the compressor was shut down and the controller power was not turned off. In this case the integral would continue to grow until the compressor was started again. To reset the integral the user just has to turn the Totalize digital constant (DCO #3) off and on again. The user should do this whenever starting a new experiment from an equilibrium starting state to purge the integral. The diagram does not include Analog Input #4, mixed temperature and Analog Output #2, and condenser water flow control valve because these were not used in the final control strategy but might be utilized in future control strategies.

The default set points programmed in the controller are set at the minimum pressure differential for the compressor to ensure the system was operating properly on start-up. These set points can then be changed either by the commissioning tool or the automatic test sequence system.

8. AUTOMATIC TEST SEQUENCE SYSTEM

An automatic test sequence system was written for the load stand using the software package VisSim Professional Edition, Version 2.0 (a printout of the program is included in Appendix D). The system reads test points from an external text file *l_stand.map* and exports them to the DX-9100 controller as the respective set points. The text file contains the set points for the suction pressure, discharge pressure, and an end variable. The purpose of the end variable is to tell the system that all test points have been obtained and to end the testing. The end variable should be zero until all tests are run at which point it should be set to one. Not only can the system change settings in the controller but it can also import certain parameters from the controller. To prevent liquid in the suction line of the compressor, the program determines the current saturated suction temperature based upon the measured suction pressure and uses this value as the suction temperature set point. If the desired suction temperature set point was automatically programmed into the controller a situation might occur when the suction temperature reaches a lower temperature than the saturation temperature. The system determines the saturation temperature by reading the current suction pressure and using the following correlation for R-22:

$$T_{sat} = \frac{-2462.33}{\ln(P_{sat}) - 15.213} - 273.13$$

where: T_{sat} = saturation temperature, °C

P_{sat} = suction pressure, kPa

The system then adds the superheat value to the saturated temperature to determine the suction temperature set point. If the difference between the correlated suction pressure and the set point suction pressure is less than 10 °C the suction pressure set point is set to the actual desired set point rather than the correlated suction pressure. By using this method of setting the suction temperature, the system is less likely to have liquid in the suction to the compressor.

To find out when the system reaches steady state, the VisSim program integrates the suction pressure, suction temperature, discharge pressure, and discharge temperature over the interval set in the program as "steady state time". It takes this value and subtracts the set point integral value. If this value is less than the tolerance value multiplied by the int time step, then steady state is satisfied for that state. The system will proceed to the next test point if all other properties are satisfied at that point also. The system imports the new test points and exports them to the controller and the procedure is repeated until all test points have been run. The variable "sim step" should be set to the step size set in the simulation setup. The variable " steady state time " should be set to the amount of time to integrate the input signals to determine steady state. "Press tolerance" should be set to the difference in the average pressure and the pressure set point which constitutes steady state. "Temp tolerance" should be set in the same manner as "Press tolerance". Under the "System Control Relays" heading there are buttons for the three relays on the system: solenoid valve, SCR, and compressor and a button to enable the integral action. By pressing these buttons the user can turn these devices on and off.

The VisSim program uses a *.dll* file which was developed using Borland C++ to communicate with the controller through the communication box. The user may have to edit this file to set the COM port to the current port being used for communications. Additional details on how to edit the *.dll* file, a printout of the VisSim program, a printout of the *.dll* source code, and further explanation on using the VisSim program are included in Appendix D.

9. CONCLUSIONS AND RECOMMENDATIONS

Using the final control strategy, the load stand achieved the desired test points in a reasonable amount of time. The data collected at the desired test points is presented in Figures 15-17.

As for recommendations, the pressure transducers used were not the ones desired due to time constraints. The pressure transducer ranges are too large for the pressures experienced for the operating conditions. The pressure rating on the tubing is around 450 psi, so the 0-1000 psi transducers are largely oversized. This introduces unnecessary errors (1% of full-scale range). 0-250 psi pressure transducers should replace the current low side pressure transducers (0-500 psi) and 0-500 psi transducers should replace the high side pressure transducers (0-1000 psi). This

would improve the accuracy of the readings being taken. In addition, when the pressure transducers were installed they were not calibrated. If the pressure transducers are replaced, this would be the ideal time to calibrate them.

The control strategy developed in the research project was for a specific compressor. As new compressors are tested the strategy will most likely need adjustment and maybe a whole new strategy will have to be developed depending upon how the system components react to the different capacities.

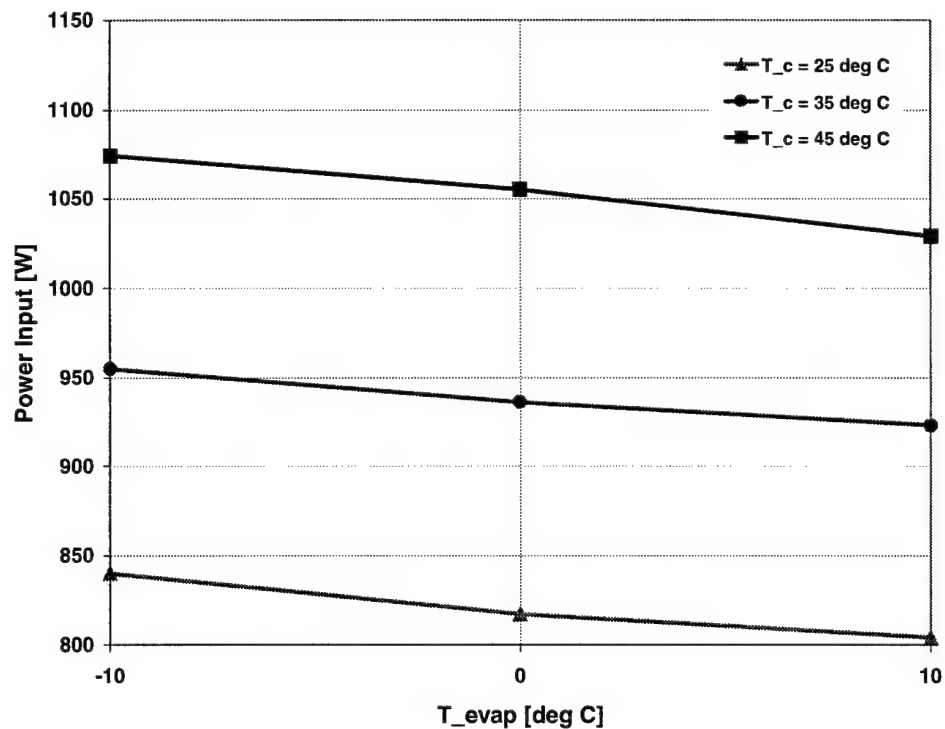


Figure 15: Power Input Measured Data

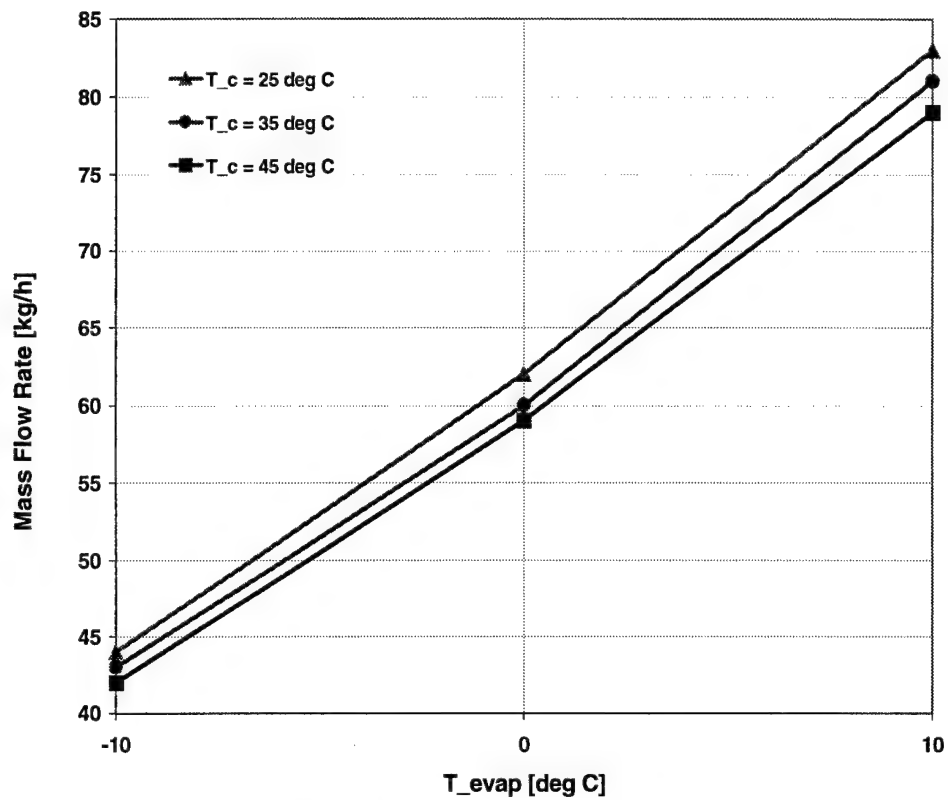


Figure 16: Mass Flow Measured Data

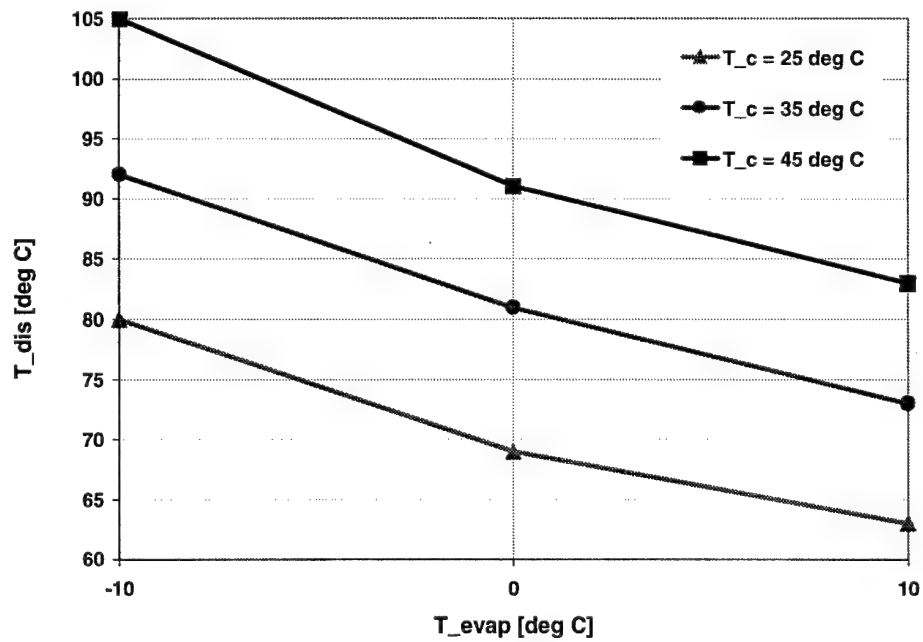
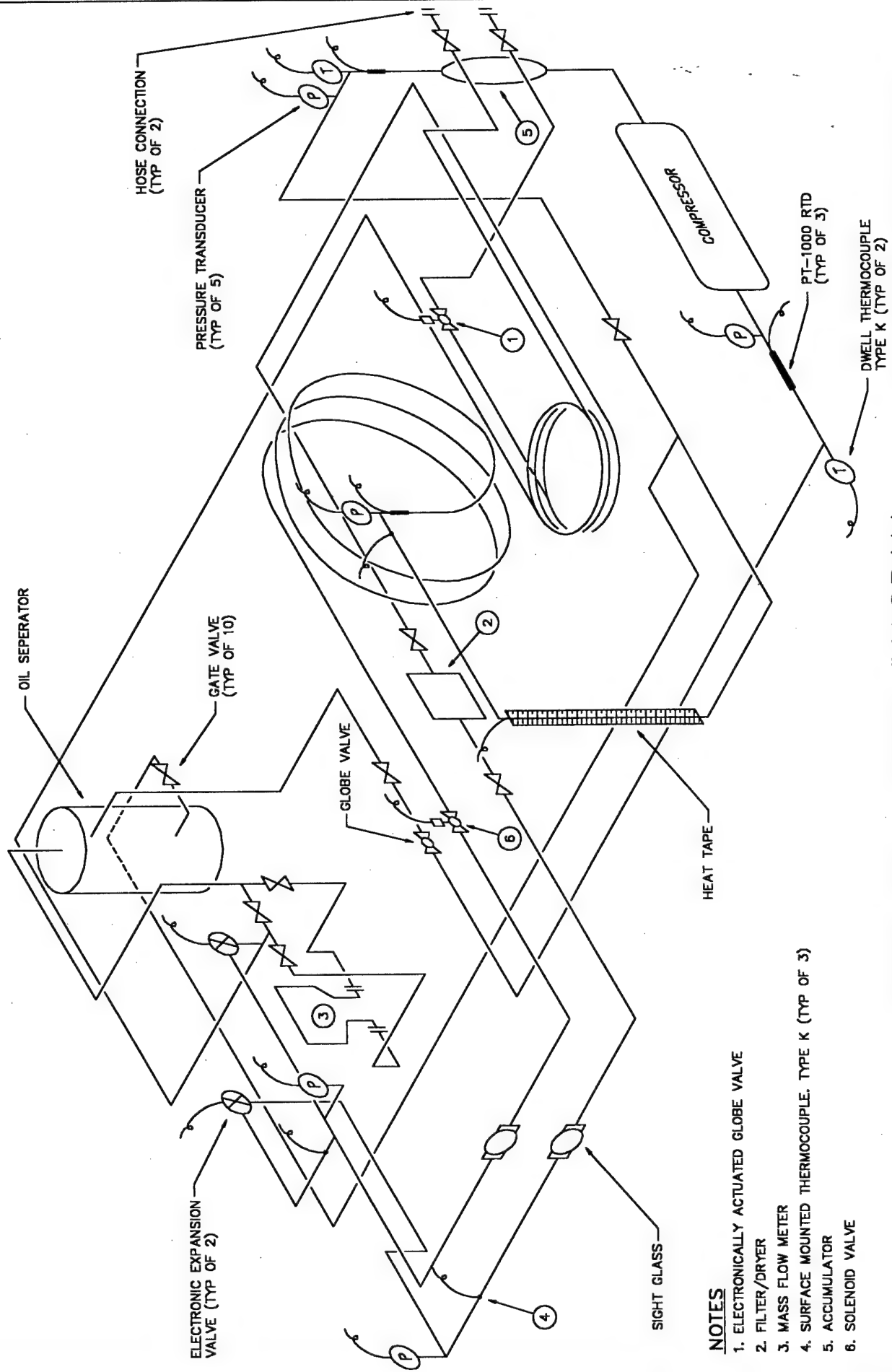


Figure 17: Discharge Temperature Measured Data

LIST OF REFERENCES

- [1] N. Halms. Mathematical Modeling of Scroll Compressors. Master's Thesis, Purdue University, 1997.
- [2] Johnson Controls, Inc. Metasys System 9100 Technical Manual. Milwaukee, Wisconsin, 1993.

APPENDIX A: SYSTEM DIAGRAMS



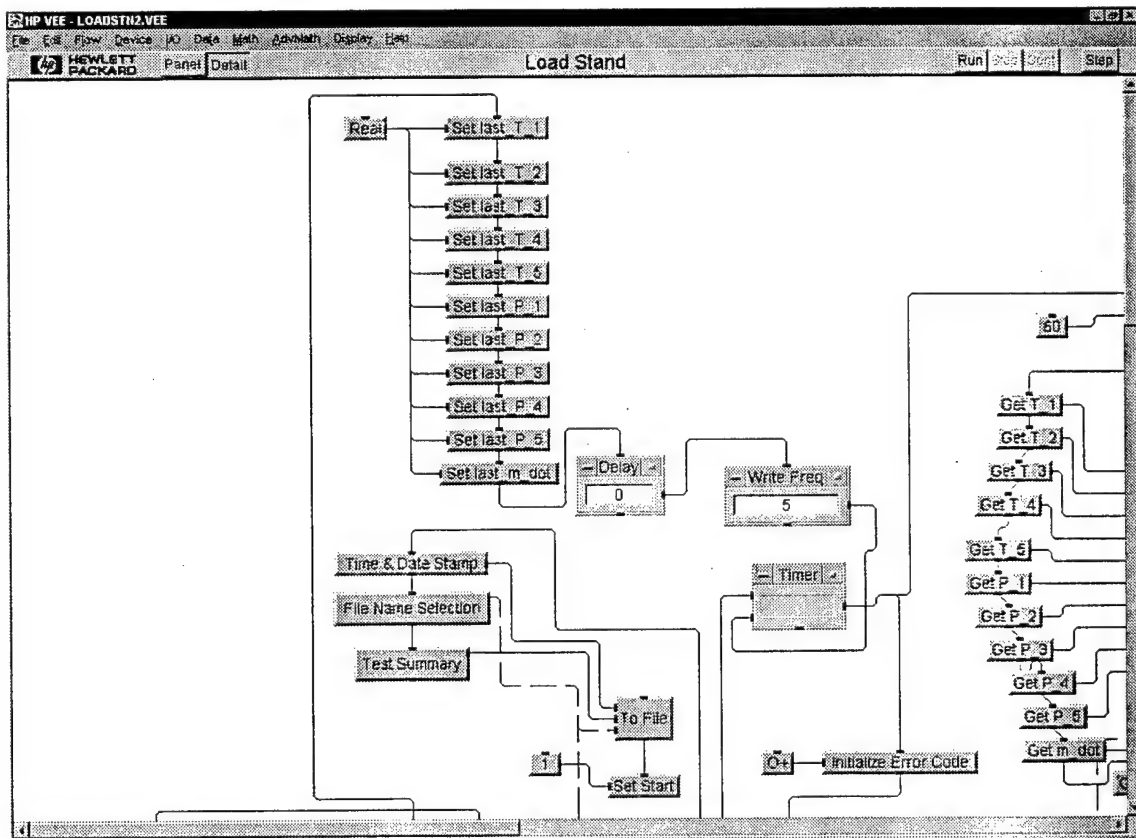
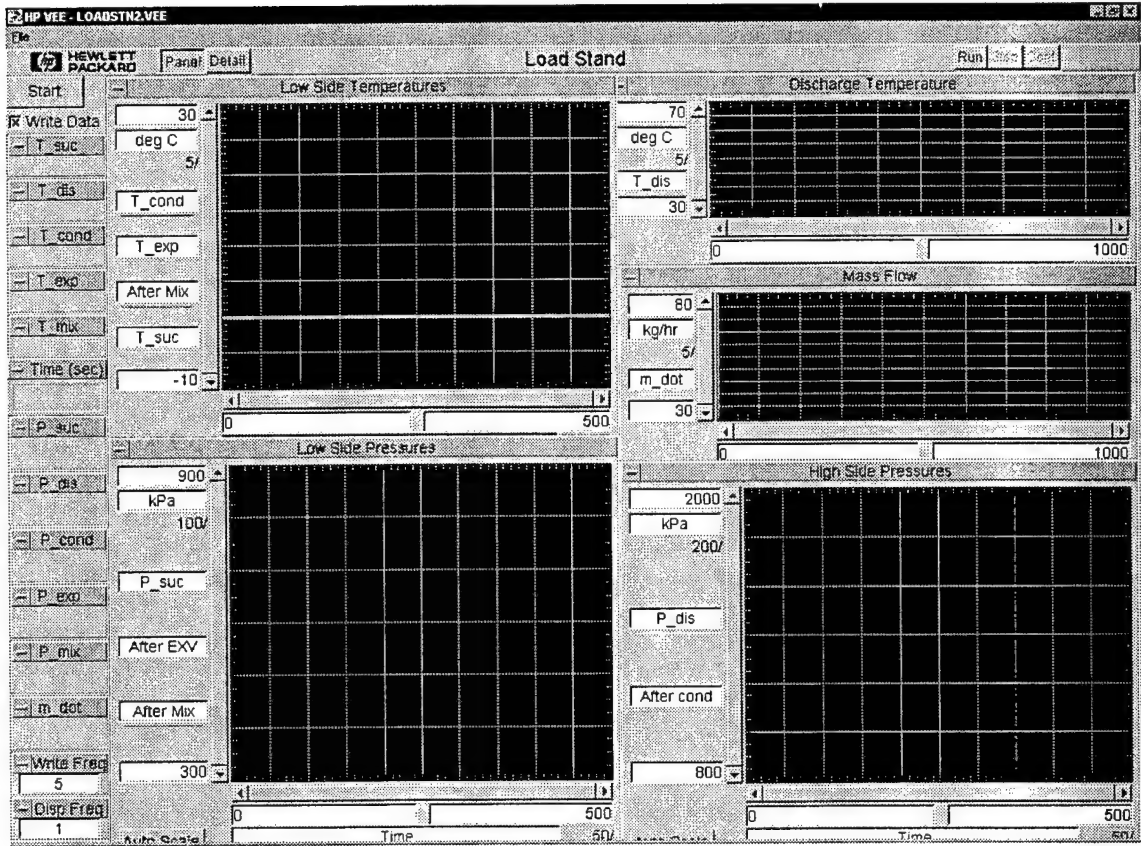
NOTES

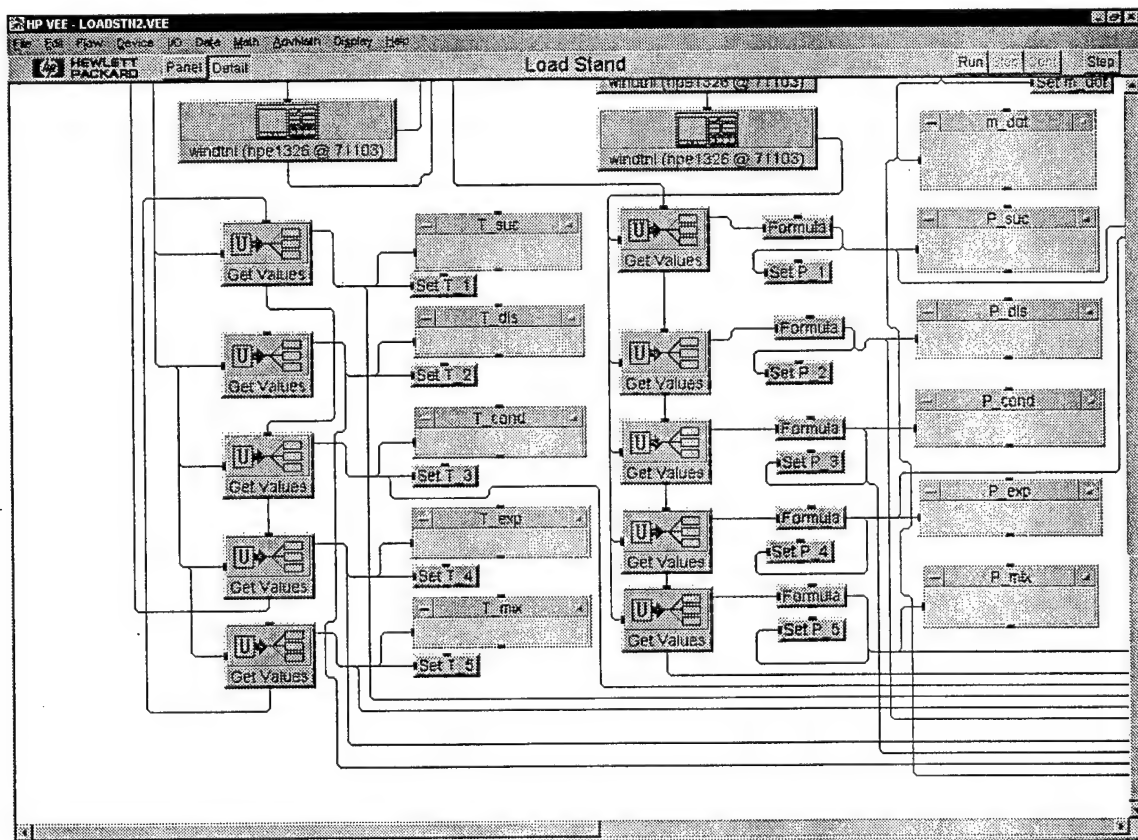
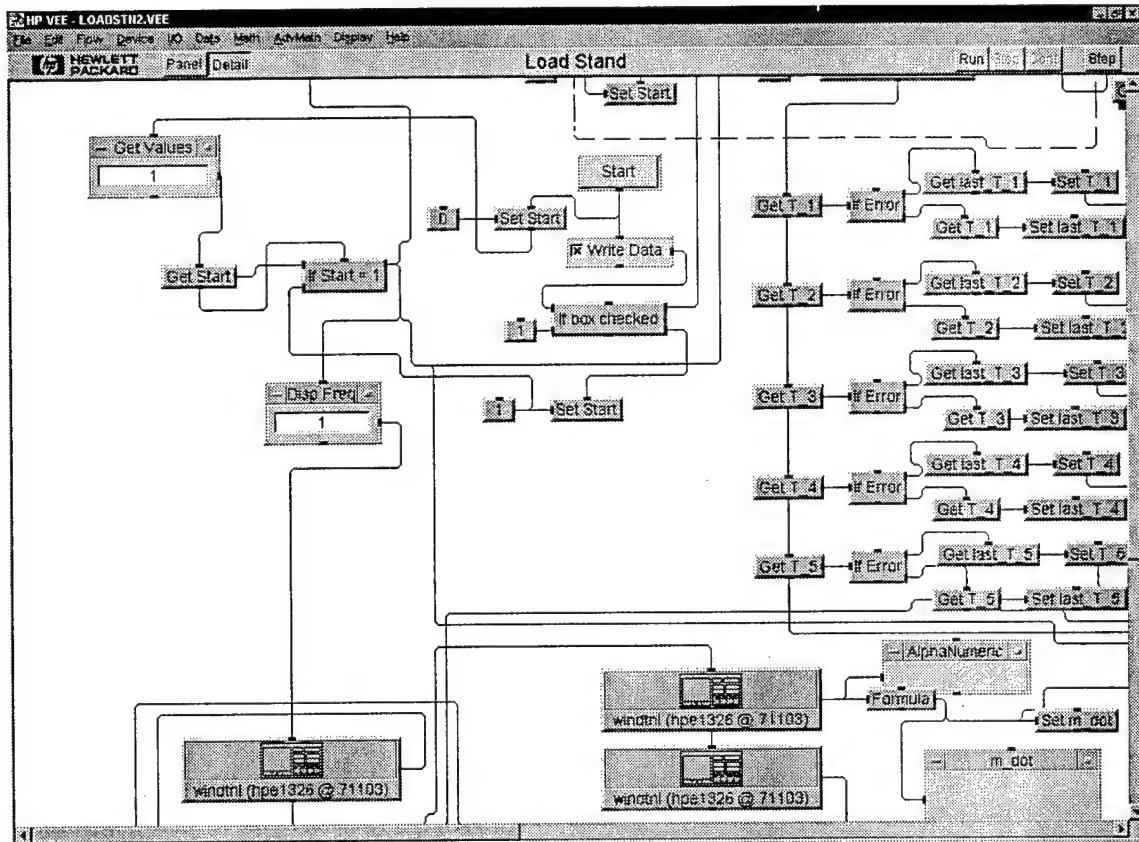
1. ELECTRONICALLY ACTUATED GLOBE VALVE
2. FILTER/DRYER
3. MASS FLOW METER
4. SURFACE MOUNTED THERMOCOUPLE. TYPE K (TYP OF 3)
5. ACCUMULATOR
6. SOLENOID VALVE

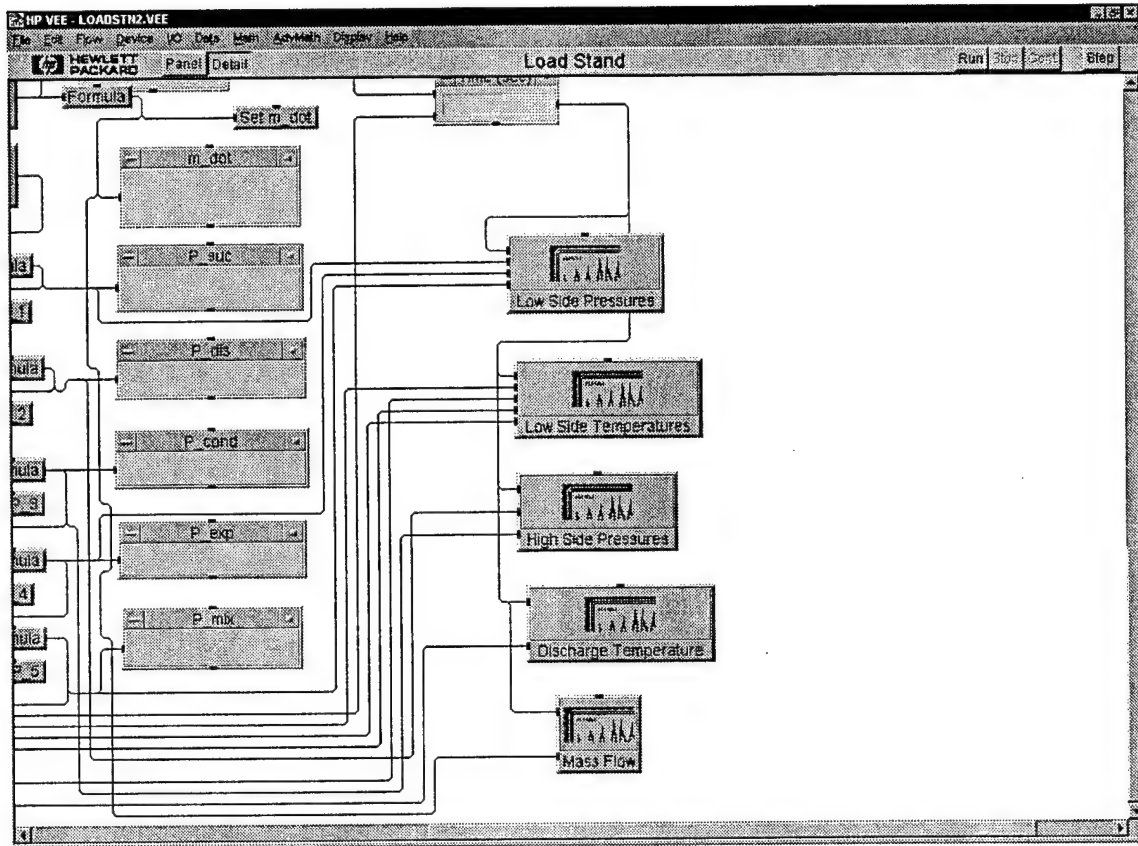
ISOMETRIC PIPING DIAGRAM

TITLE: LOAD STAND ISOMETRIC PIPING DIAGRAM	FILENAME: ISOMETRIC.DWG
DESIGNED BY: NILS HALM	SCALE: NO SCALE
DRAWN BY: DREW CAUSEY	SHEET 1 OF 2
DATE: 29 JUL 97	

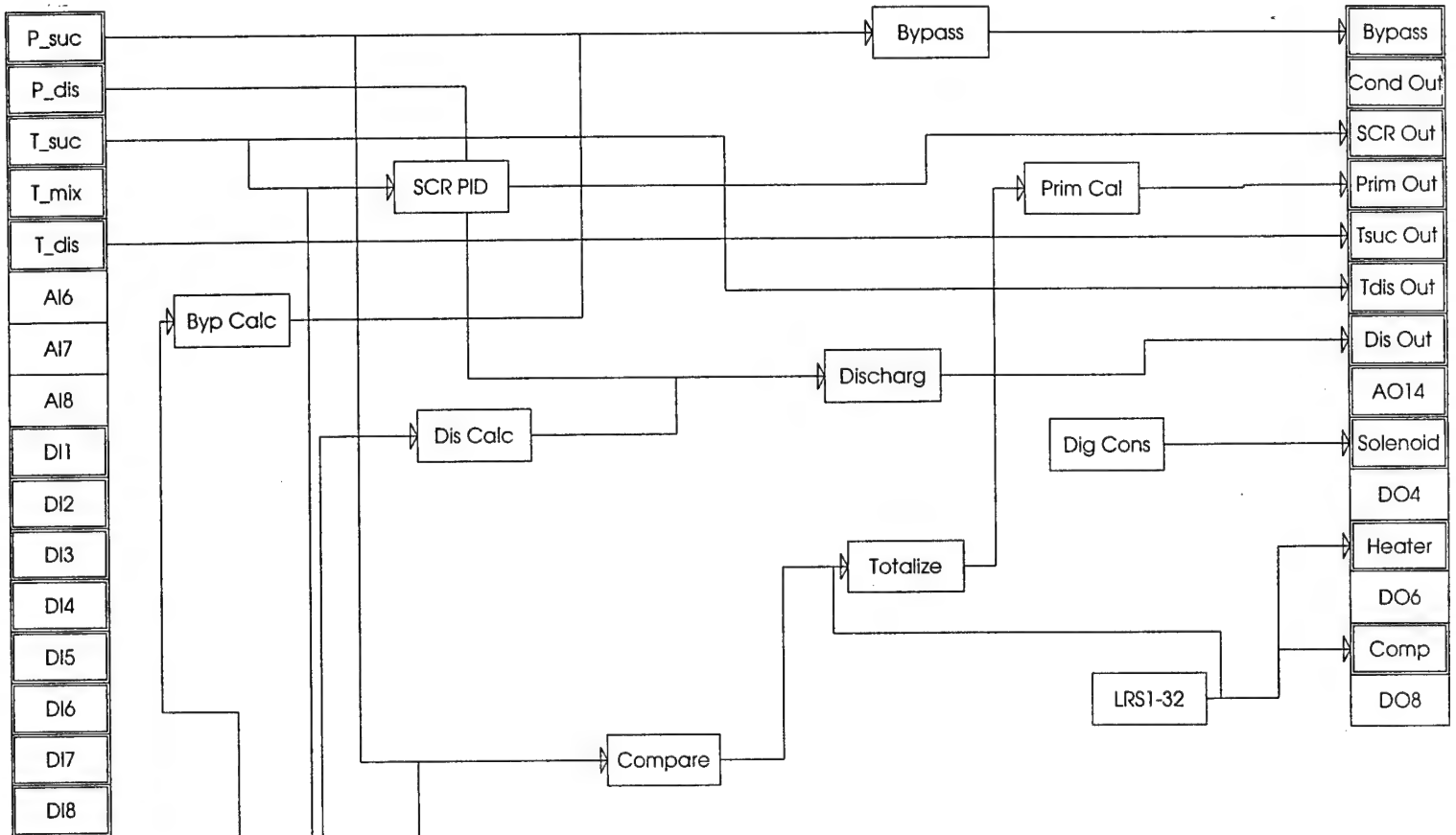
APPENDIX B: HP VEE PROGRAM CODE







APPENDIX C: CONTROLLER CONFIGURATION



PLC #2

Relays

PM9
 PM10
 PM11
 PLC3

ANA IN (ACT) (AI1) - Data

User Name :P_suc
Description :P_suc

Measurement Units	0
Type of Active Input	0
High Range	6209.0000
Low Range	-685.0000
High Limit	6500.0000
Low Limit	-700.0000
Filter Constant (sec)	0.0000
Limit Differential	0.0000
Alarm Unfiltered (0=N)	0
Square Root (0=N)	0

ANA IN (ACT) (AI1) - Output-Tag

AI	P_suc	AIL
AI%		OVR
AIH		UNR

ANA IN (ACT) (AI2) - Data

User Name :P_dis
Description :P_dis

Measurement Units	0
Type of Active Input	0
High Range	12410.5996
Low Range	-1378.9600
High Limit	13000.0000
Low Limit	-1400.0000
Filter Constant (sec)	0.0000
Limit Differential	0.0000
Alarm Unfiltered (0=N)	0
Square Root (0=N)	0

ANA IN (ACT) (AI2) - Output-Tag

AI	P_dis	AIL
AI%		OVR
AIH		UNR

ANA IN (PASS) (AI3) - Data

User Name :T_suc
Description :T_suc

Type of Passive Input	4
High Range (Control)	200.0000
Low Range (Control)	-50.0000
High Limit	250.0000
Low Limit	-75.0000
Filter Constant (sec)	0.0000
Limit Differential	2.0000
Alarm Unfiltered (0=N)	0

ANA IN (PASS) (AI3) - Output-Tag

AI	T_suc	AIL
AI%		OVR
AIH		UNR

ANA IN (PASS) (AI4) - Data

User Name :T_mix
Description :T_mix

Type of Passive Input	4
High Range (Control)	200.0000
Low Range (Control)	-50.0000
High Limit	250.0000
Low Limit	-100.0000
Filter Constant (sec)	0.0000
Limit Differential	2.0000
Alarm Unfiltered (0=N)	0

ANA IN (PASS) (AI4) - Output-Tag

AI	T_mix	AIL
AI%		OVR
AIH		UNR

ANA IN (PASS) (AI5) - Data

User Name :T_dis
Description :T_dis

Type of Passive Input	4
High Range (Control)	200.0000
Low Range (Control)	-50.0000
High Limit	250.0000
Low Limit	-100.0000
Filter Constant (sec)	0.0000
Limit Differential	2.0000
Alarm Unfiltered (0=N)	0

ANA IN (PASS) (AI5) - Output-Tag

AI	T_dis	AIL
AI%		OVR
AIH		UNR

DIGITAL IN (DI1) - Data

User Name :
Description :

Prescaler (counts)	1
--------------------	---

DIGITAL IN (DI1) - Output-Tag

DI	DIC
----	-----

DIGITAL IN (DI2) - Data

User Name :
Description :

Prescaler (counts)	1
--------------------	---

DIGITAL IN (DI2) - Output-Tag

DI	DIC
----	-----

DIGITAL IN (DI3) - Data

User Name :
Description :

Prescaler (counts) 1

DIGITAL IN (DI3) - Output-Tag

DI

DIC

DIGITAL IN (DI4) - Data

User Name :
Description :

Prescaler (counts) 1

DIGITAL IN (DI4) - Output-Tag

DI

DIC

DIGITAL IN (DI5) - Data

User Name :
Description :

Prescaler (counts) 1

DIGITAL IN (DI5) - Output-Tag

DI

DIC

DIGITAL IN (DI6) - Data

User Name :
Description :

Prescaler (counts) 1

DIGITAL IN (DI6) - Output-Tag

DI

DIC

DIGITAL IN (DI7) - Data

User Name :
Description :

Prescaler (counts) 1

DIGITAL IN (DI7) - Output-Tag

DI

DIC

DIGITAL IN (DI8) - Data

User Name :
Description :

Prescaler (counts) 1

DIGITAL IN (DI8) - Output-Tag

DI

DIC

ANALOG OUT (AO1) - Data

User Name :Bypass
Description :Bypass EXV

Type of Output	1	Low Range	0.0000
Source Point	--> Byp OCM	High Limit (%)	100.0000
Output Forcing	-->	Low Limit (%)	10.0000
Enable Limits	-->	Forcing Level (%)	0.0000
Increase Source	-->	Hold on Powerup(0=N)	0
Decrease Source	-->	Auto on Powerup(0=N)	1
High Range	100.0000		

ANALOG OUT (AO1) - Input-Tag

AO@	Byp OCM	INC@
AOF@		DEC@
ENL@		

ANALOG OUT (AO1) - Output-Tag

OUT	Byp Out	AOL
OUH		AOF
AOH		

ANALOG OUT (AO2) - Data

User Name :Cond Out
Description :Condenser Flow

Type of Output	1	Low Range	0.0000
Source Point	-->	High Limit (%)	100.0000
Output Forcing	-->	Low Limit (%)	0.0000
Enable Limits	-->	Forcing Level (%)	0.0000
Increase Source	-->	Hold on Powerup(0=N)	0
Decrease Source	-->	Auto on Powerup(0=N)	1
High Range	100.0000		

ANALOG OUT (AO2) - Input-Tag

AO@	INC@
AOF@	DEC@
ENL@	

ANALOG OUT (AO2) - Output-Tag

OUT	Cond Out	AOL
OUH		AOF
AOH		

ANALOG OUT (AO9) - Data

User Name :SCR Out
Description :SCR Analog Output

Type of Output	3	Low Range	0.0000
Source Point	--> SCR OCM	High Limit (%)	100.0000
Output Forcing	-->	Low Limit (%)	0.0000
Enable Limits	-->	Forcing Level (%)	0.0000
Increase Source	-->	Hold on Powerup(0=N)	0
Decrease Source	-->	Auto on Powerup(0=N)	1
High Range	100.0000		

ANALOG OUT (AO9) - Input-Tag

AO@	SCR OCM	INC@
AOF@		DEC@
ENL@		

ANALOG OUT (AO9) - Output-Tag

OUT	SCR Out	AOL
OUH		AOF
AOH		

ANALOG OUT (AO10) - Data

User Name :Prim Out
Description :Primary EXV

Type of Output	1	Low Range	0.0000
Source Point	--> Prim OCM	High Limit (%)	100.0000
Output Forcing	-->	Low Limit (%)	0.0000
Enable Limits	-->	Forcing Level (%)	0.0000
Increase Source	-->	Hold on Powerup(0=N)	0
Decrease Source	-->	Auto on Powerup(0=N)	1
High Range	100.0000		

ANALOG OUT (AO10) - Input-Tag

AO@	Prim OCM	INC@
AOF@		DEC@
ENL@		

ANALOG OUT (AO10) - Output-Tag

OUT	Prim Out	AOL
OUH		AOF
AOH		

ANALOG OUT (AO11) - Data

User Name :Tsuc Out
Description :T_suc Display

Type of Output	1	Low Range	-50.0000
Source Point	--> T_suc	High Limit (%)	100.0000
Output Forcing	-->	Low Limit (%)	0.0000
Enable Limits	-->	Forcing Level (%)	0.0000
Increase Source	-->	Hold on Powerup(0=N)	0
Decrease Source	-->	Auto on Powerup(0=N)	0
High Range	200.0000		

ANALOG OUT (AO11) - Input-Tag

AO@	T_suc	INC@
AOF@		DEC@
ENL@		

ANALOG OUT (AO11) - Output-Tag

OUT		AOL
OUH		AOF
AOH		

ANALOG OUT (AO12) - Data

User Name :Tdis Out
Description :T_dis Display

Type of Output	1	Low Range	-50.0000
Source Point	--> T_dis	High Limit (%)	100.0000
Output Forcing	-->	Low Limit (%)	0.0000
Enable Limits	-->	Forcing Level (%)	0.0000
Increase Source	-->	Hold on Powerup(0=N)	0
Decrease Source	-->	Auto on Powerup(0=N)	0
High Range	200.0000		

ANALOG OUT (AO12) - Input-Tag

AO@	T_dis	INC@
AOF@		DEC@
ENL@		

ANALOG OUT (AO12) - Output-Tag

OUT	AOL
OUH	AOF
AOH	

ANALOG OUT (AO13) - Data

User Name :Dis Out
Description :Discharge Output

Type of Output	1	Low Range	0.0000
Source Point	--> Dis OCM	High Limit (%)	100.0000
Output Forcing	-->	Low Limit (%)	0.0000
Enable Limits	-->	Forcing Level (%)	0.0000
Increase Source	-->	Hold on Powerup(0=N)	0
Decrease Source	-->	Auto on Powerup(0=N)	0
High Range	100.0000		

ANALOG OUT (AO13) - Input-Tag

AO@	Dis OCM	INC@
AOF@		DEC@
ENL@		

ANALOG OUT (AO13) - Output-Tag

OUT	Dis Out	AOL
OUH		AOF
AOH		

ANALOG OUT (AO14) - Data

User Name :
Description :

Type of Output	0	Low Range	0
Source Point	-->	High Limit (%)	100
Output Forcing	-->	Low Limit (%)	0
Enable Limits	-->	Forcing Level (%)	0
Increase Source	-->	Hold on Powerup(0=N)	0
Decrease Source	-->	Auto on Powerup(0=N)	0
High Range	100		

ANALOG OUT (AO14) - Input-Tag

AO@
AOF@
ENL@

INC@
DEC@

ANALOG OUT (AO14) - Output-Tag

OUT
OUH
AOH

AOL
AOF

ON/OFF (DO3) - Data

User Name :Solenoid
Description :Solenoid Valve

Source Point --> Solenoid

ON/OFF (DO3) - Input-Tag

DO@ Solenoid

ON/OFF (DO3) - Output-Tag

DO Sol Out OUH

ON/OFF (DO5) - Data

User Name :Heater
Description :Heat Tape

Source Point --> Heater

ON/OFF (DO5) - Input-Tag

DO@ Heater

ON/OFF (DO5) - Output-Tag

DO SCR Out OUH

ON/OFF (DO7) - Data

User Name :Comp
Description :Compressor

Source Point --> Comp

ON/OFF (DO7) - Input-Tag

DO@ Comp

ON/OFF (DO7) - Output-Tag

DO Comp Out OUH

PID (PID1) - Data

User Name :Bypass
Description :Suction Pressure PID

Ena Shutoff: 0=N	0	Maximum WSP	-->
Shutoff Out Level	0.0000	Local Set Pt.(LSP)	0.0000
Ena Startup: 0=N	1	Proport. Band(PB)	-6.0000
Startup Out Level	50.0000	Reset Action(TI)	5.0000
Ena Symm Mode: 0=N	0	Rate Action(TD)	0.0000
ExtForce Out Level	0.0000	Standby Bias(BSB)	0.0000
Ena PID to P: 0=N	0	Off Mode Bias(BOF)	0.0000
Remote Mode: 0=N	1	Symmetry Band(SBC)	0.0000
Ena OFF Trans: 0=N	0	Err Deadband(EDB)	0.0000
Process Variable --> P_suc		Output Bias(OB)	0.0000
Remote Setpoint --> Byp WSP		Out High Lmt(HIL)	100.0000
Reference Variable-->		Out Low Lmt(LOL)	10.0000
Proportional Band -->		Dev H.H.Limit(DHH)	300.0000
OFF Mode Control -->		Dev High Limit(DH)	115.0000
Standby Control -->		Dev Low Limit(DL)	115.0000
Reverse Action -->		Dev L.L.Limit(DLL)	300.0000
External Forcing -->		Minimum WSP(MNWS)	200.0000
Output Bias -->		Maximum WSP(MXWS)	1200.0000
Minimum WSP -->			

PID (PID1) - Input-Tag

PV@	P_suc	RA@
RS@	Byp LSP	EF@
RV@		OB@
PB@		MNWS@
OF@		MXWS@
SB@		

PID (PID1) - Output-Tag

OCM	Byp OCM	EF
WSP		LLDA
LSP		LDA
HLD		HDA
CMF		HHDA
SOF		CML
STA		CMH

PID (PID2) - Data

User Name :Discharg
Description :Discharge Pressure PID

Ena Shutoff: 0=N	0	Maximum WSP	-->
Shutoff Out Level	0.0000	Local Set Pt.(LSP)	0.0000
Ena Startup: 0=N	0	Proport. Band(PB)	6.0000
Startup Out Level	100.0000	Reset Action(TI)	5.0000
Ena Symm Mode: 0=N	0	Rate Action(TD)	0.0000
ExtForce Out Level	0.0000	Standby Bias(BSB)	0.0000
Ena PID to P: 0=N	0	Off Mode Bias(BOF)	0.0000
Remote Mode: 0=N	1	Symmetry Band(SBC)	0.0000
Ena OFF Trans: 0=N	0	Err Deadband(EDB)	0.0000
Process Variable --> P_dis		Output Bias(OB)	0.0000
Remote Setpoint --> Dis LSP		Out High Lmt(HIL)	100.0000
Reference Variable-->		Out Low Lmt(LOL)	0.0000
Proportional Band -->		Dev H.H.Limit(DHH)	500.0000
OFF Mode Control -->		Dev High Limit(DH)	50.0000
Standby Control -->		Dev Low Limit(DL)	50.0000
Reverse Action -->		Dev L.L.Limit(DLL)	500.0000
External Forcing -->		Minimum WSP(MNWS)	400.0000
Output Bias -->		Maximum WSP(MXWS)	3000.0000
Minimum WSP -->			

PID (PID2) - Input-Tag

PV@	P_dis	RA@
RS@		EF@
RV@		OB@
PB@		MNWS@
OF@		MXWS@
SB@		

PID (PID2) - Output-Tag

OCM	Dis OCM	EF
WSP		LLDA
LSP		LDA
HLD		HDA
CMP		HHDA
SOF		CML
STA		CMH

PID (PID3) - Data

User Name :SCR PID
Description :Suction Temp PID

Ena Shutoff: 0=N	0	Maximum WSP	-->
Shutoff Out Level	0.0000	Local Set Pt.(LSP)	0.0000
Ena Startup: 0=N	0	Proport. Band(PB)	-15.0000
Startup Out Level	0.0000	Reset Action(TI)	1.0000
Ena Symm Mode: 0=N	0	Rate Action(TD)	0.0000
ExtForce Out Level	0.0000	Standby Bias(BSB)	0.0000
Ena PID to P: 0=N	1	Off Mode Bias(BOF)	0.0000
Remote Mode: 0=N	1	Symmetry Band(SBC)	0.0000
Ena OFF Trans: 0=N	0	Err Deadband(EDB)	10.0000
Process Variable --> T_suc		Output Bias(OB)	0.0000
Remote Setpoint --> SCR LSP		Out High Lmt(HIL)	40.0000
Reference Variable-->		Out Low Lmt(LOL)	0.0000
Proportional Band -->		Dev H.H.Limit(DHH)	100.0000
OFF Mode Control -->		Dev High Limit(DH)	5.0000
Standby Control -->		Dev Low Limit(DL)	5.0000
Reverse Action -->		Dev L.L.Limit(DLL)	100.0000
External Forcing -->		Minimum WSP(MNWS)	-20.0000
Output Bias -->		Maximum WSP(MXWS)	250.0000
Minimum WSP -->			

PID (PID3) - Input-Tag

PV@	T_suc	RA@
RS@		EF@
RV@		OB@
PB@		MNWS@
OF@		MXWS@
SB@		

PID (PID3) - Output-Tag

OCM	SCR OCM	EF
WSP		LLDA
LSP	SCR LSP	LDA
HLD		HDA
CMP		HHDA
SOF		CML
STA		CMH

CALC (CALC4) - Data

User Name :Prim Cal
Description :

Input#n	=====	Connection	=====	K(n)
#0		:::~:::		0.0000
#1	-->	Prim Int		0.1000
#2	-->			0.0000
#3	-->			0.0000
#4	-->			0.0000
#5	-->			0.0000
#6	-->			0.0000
#7	-->			0.0000
#8	-->			1.0000
#9		:::~:::		0.0000
High Limit	20.0000			:::~:::
Low Limit	3.0000			:::~:::
Egn (1or2)	1			:::~:::

 CALC (CALC4) - Input-Tag

I1@	Prim Int	I5@
I2@		I6@
I3@		I7@
I4@		I8@

 CALC (CALC4) - Output-Tag

NCM	Prim OCM	NML
HLD		NMH

 COMPARATOR. (COMPRTR5) - Data

User Name :Compare
 Description :High/Low Cut-Offs

CHANNEL TYPE #1	1	CHANNEL TYPE #5	3
Analog Input #1	--> P_dis	Analog Input #5	-->
Set Point #1	-->	Set Point #5	-->
Set Point Value #1	2500.0000	Set Point Value #5	0.0000
Differential #1	1000.0000	Differential #5	0.0000
CHANNEL TYPE #2	1	CHANNEL TYPE #6	2
Analog Input #2	--> T_dis	Analog Input #6	-->
Set Point #2	-->	Set Point #6	-->
Set Point Value #2	110.0000	Set Point Value #6	-0.5000
Differential #2	30.0000	Differential #6	0.0000
CHANNEL TYPE #3	2	CHANNEL TYPE #7	0
Analog Input #3	--> P_suc	Analog Input #7	-->
Set Point #3	-->	Set Point #7	-->
Set Point Value #3	100.0000	Set Point Value #7	0.0000
Differential #3	500.0000	Differential #7	2.0000
CHANNEL TYPE #4	1	CHANNEL TYPE #8	0
Analog Input #4	--> T_suc	Analog Input #8	-->
Set Point #4	--> Prim LSP	Set Point #8	-->
Set Point Value #4	0.0000	Set Point Value #8	0.0000
Differential #4	0.0000	Differential #8	2.0000

 COMPARATOR. (COMPRTR5) - Input-Tag

I1@	P_dis	I5@
SP1@		SP5@
I2@	T_dis	I6@
SP2@		SP6@
I3@	P_suc	I7@
SP3@		SP7@
I4@	T_suc	I8@
SP4@		SP8@

 COMPARATOR. (COMPRTR5) - Output-Tag

NCM1		NCM5	Cond Dif
LS1	Hi Press	LS5	
HLD1		HLD5	
NCM2		NCM6	
LS2	Hi Temp	LS6	
HLD2		HLD6	
NCM3		NCM7	
LS3	Lo Press	LS7	
HLD3		HLD7	
NCM4	Prim dif	NCM8	
LS4		LS8	
HLD4		HLD8	

 CALC (CALC6) - Data

User Name :Byp Calc
 Description :Bypass LSP Calculator

Input#n	=====	Connection	=====	K(n)
#0		:::::		0.0000
#1	-->	Byp LSP1		100.0000
#2	-->	Byp LSP2		1.0000
#3	-->			0.0000
#4	-->			0.0000
#5	-->			0.0000
#6	-->			0.0000
#7	-->			0.0000
#8	-->			1.0000
#9		:::::		0.0000
High Limit		1000.0000		:::::
Low Limit		0.0000		:::::
Eqn (1or2)		1		:::::

 CALC (CALC6) - Input-Tag

I1@	Byp LSP1	I5@
I2@	Byp LSP2	I6@
I3@		I7@
I4@		I8@

 CALC (CALC6) - Output-Tag

NCM	Byp WSP	NML
HLD		NMH

 TOTALIZATION (TOTAL7) - Data

User Name :Totalize
 Description :Primary Integral Action

TOTALIZATION 1 TYPE	2	TOTALIZATION 5 TYPE	0
Source #1	--> Prim dif	Source #5	-->
Reset #1	--> Reset	Reset #5	-->
Full Scale Limit #1	200.0000	Full Scale Limit #5	1000.0000
Scale/Time Const.#1	1.0000	Scale/Time Const.#5	1.0000
Incrmnt Acc.#1(0=N)	0	Incrmnt Acc.#5(0=N)	0
TOTALIZATION 2 TYPE	0	TOTALIZATION 6 TYPE	0
Source #2	-->	Source #6	-->
Reset #2	-->	Reset #6	-->
Full Scale Limit #2	1000.0000	Full Scale Limit #6	1000.0000
Scale/Time Const.#2	1.0000	Scale/Time Const.#6	1.0000
Incrmnt Acc.#2(0=N)	0	Incrmnt Acc.#6(0=N)	0
TOTALIZATION 3 TYPE	0	TOTALIZATION 7 TYPE	0
Source #3	-->	Source #7	-->
Reset #3	-->	Reset #7	-->
Full Scale Limit #3	0.0000	Full Scale Limit #7	1000.0000
Scale/Time Const.#3	1.0000	Scale/Time Const.#7	1.0000
Incrmnt Acc.#3(0=N)	0	Incrmnt Acc.#7(0=N)	0
TOTALIZATION 4 TYPE	0	TOTALIZATION 8 TYPE	0
Source #4	-->	Source #8	-->
Reset #4	-->	Reset #8	-->
Full Scale Limit #4	1000.0000	Full Scale Limit #8	1000.0000
Scale/Time Const.#4	1.0000	Scale/Time Const.#8	1.0000
Incrmnt Acc.#4(0=N)	0	Incrmnt Acc.#8(0=N)	0

TOTALIZATION (TOTAL7) - Input-Tag

I1@	Prim dif	I5@
RS1@	Reset	RS5@
I2@		I6@
RS2@		RS6@
I3@		I7@
RS3@		RS7@
I4@		I8@
RS4@		RS8@

TOTALIZATION (TOTAL7) - Output-Tag

TOT1	Prim Int	TOT5
FSS1		FSS5
HLD1		HLD5
TOT2		TOT6
FSS2		FSS6
HLD2		HLD6
TOT3		TOT7
FSS3		FSS7
HLD3		HLD7
TOT4		TOT8
FSS4		FSS8
HLD4		HLD8

CALC (CALC8) - Data

User Name :Dis Calc
Description :Disch LSP Calculator

Input#n	===== Connection	===== K(n)
#0	:::::	0.0000
#1	--> Dis LSP1	100.0000
#2	--> Dis LSP2	1.0000
#3	-->	0.0000
#4	-->	0.0000
#5	-->	0.0000
#6	-->	0.0000
#7	-->	0.0000
#8	-->	1.0000
#9	:::::	0.0000
High Limit	3000.0000	:::::
Low Limit	0.0000	:::::
Eqn (lor2)	1	:::::

CALC (CALC8) - Input-Tag

I1@	Dis LSP1	I5@
I2@	Dis LSP2	I6@
I3@		I7@
I4@		I8@

CALC (CALC8) - Output-Tag

NCM	Dis LSP	NML
HLD		NMH

ANALOG CONST (ACO) - Data

User Name :A Const
Description :Analog Constants

ACO #1	0.0000
ACO #2	0.0000
ACO #3	0.0000
ACO #4	0.0000
ACO #5	0.0000
ACO #6	0.0000
ACO #7	0.0000
ACO #8	0.0000

ANALOG CONST (ACO) - Output-Tag

ACO1	Byp LSP1	ACO5	Dis LSP2
ACO2	Dis LSP1	ACO6	Byp LSP2
ACO3	SCR LSP	ACO7	
ACO4	Prim LSP	ACO8	

DIGITAL CONST (DCO) - Data

User Name :Dig Cons
 Description :Digital Constants

DCO #1	0	DCO #17	0
DCO #2	0	DCO #18	0
DCO #3	1	DCO #19	0
DCO #4	1	DCO #20	0
DCO #5	1	DCO #21	0
DCO #6	0	DCO #22	0
DCO #7	1	DCO #23	0
DCO #8	0	DCO #24	0
DCO #9	0	DCO #25	0
DCO #10	0	DCO #26	0
DCO #11	0	DCO #27	0
DCO #12	0	DCO #28	0
DCO #13	0	DCO #29	0
DCO #14	0	DCO #30	0
DCO #15	0	DCO #31	0
DCO #16	0	DCO #32	0

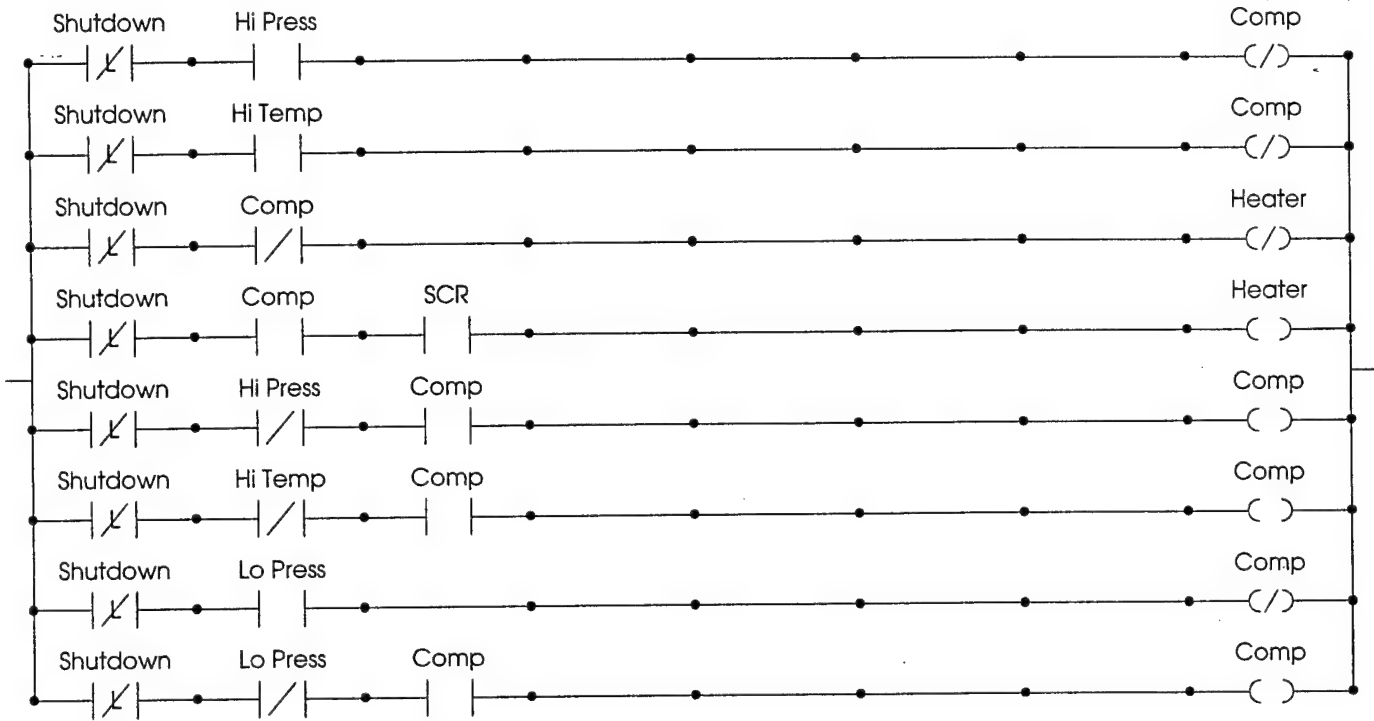
DIGITAL CONST (DCO) - Output-Tag

DC01	Shutdown	DC017
DC02	Totalize	DC018
DC03	Solenoid	DC019
DC04	SCR	DC020
DC05	Comp	DC021
DC06		DC022
DC07		DC023
DC08		DC024
DC09		DC025
DC010		DC026
DC011		DC027
DC012		DC028
DC013		DC029
DC014		DC030
DC015		DC031
DC016		DC032

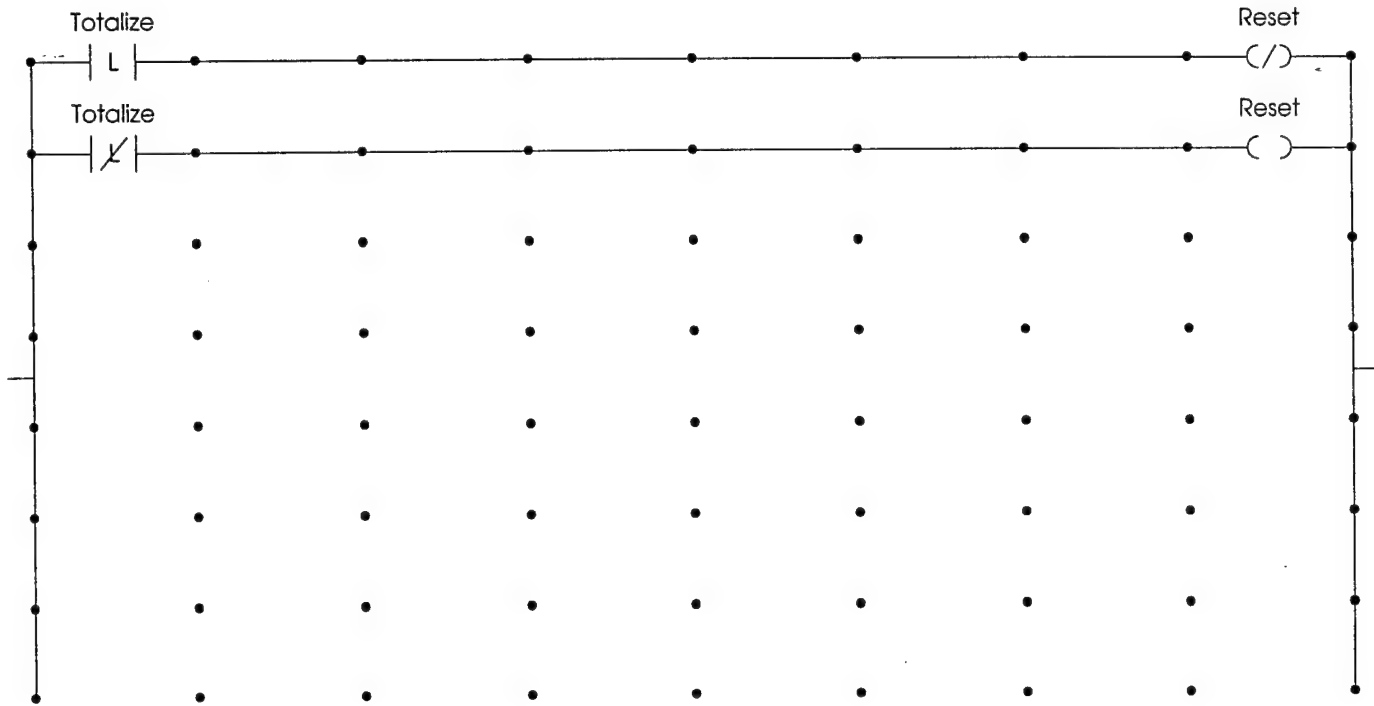
Logic Module Ladder Diagram - PLC1

User Name : Relays

Description: Relay control structure



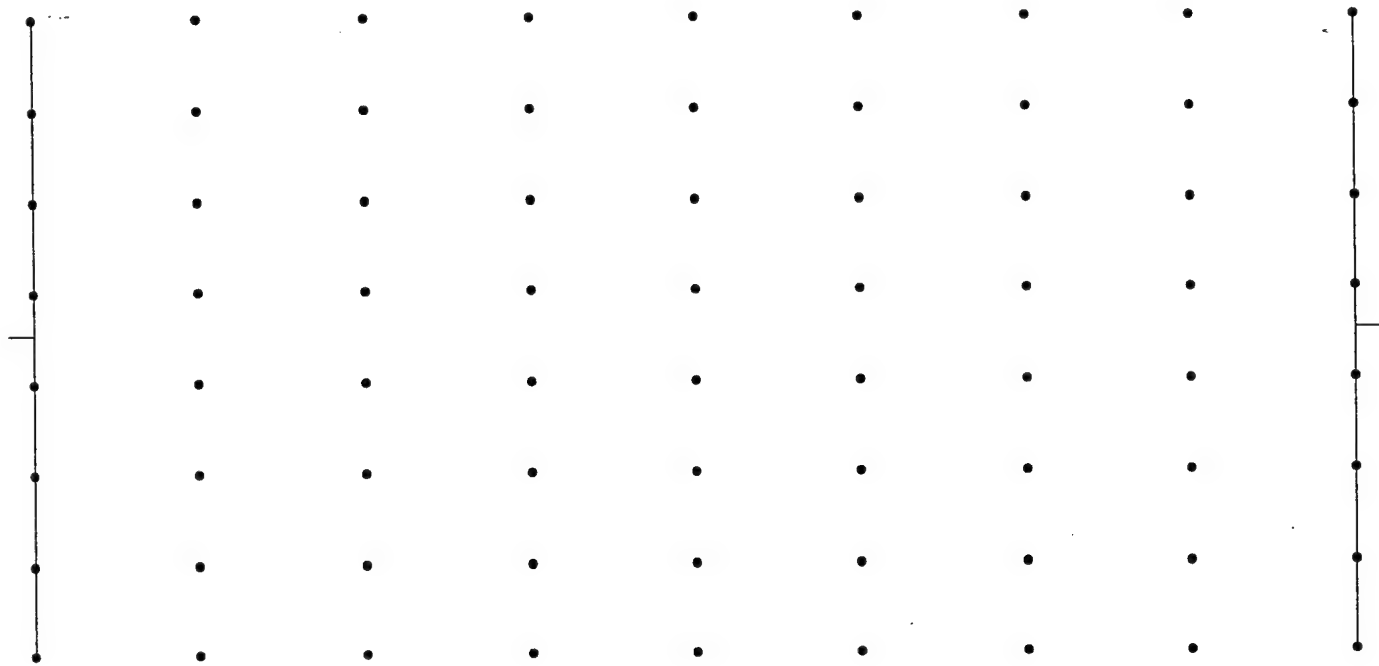
Logic Module Ladder Diagram - PLC2
User Name : PLC #2
Description: Misc Controls



Logic Module Ladder Diagram - PLC3

User Name :

Description:



* GX9100 Version 3.00 Model DDL Source File for C:\LOADST-1\LOADSTND.DXS

```

@MODEL+
CSMODEL "DX9100", "DX9100"
***** GROUP TITLES *****
AITITLE "Analog Inputs"
AOTITLE "Analog Outputs"
ADTITLE "Analog Data"
BITITLE "Binary Inputs"
BOTITLE "Binary Outputs"
BDTITLE "Binary Data"
***** INPUTS *****
* LONGNAME "P_suc"
CSAI "AI1",N,N,"P_suc","kPa"
* LONGNAME "P_dis"
CSAI "AI2",N,N,"P_dis","kPa"
* LONGNAME "T_suc"
CSAI "AI3",N,N,"T_suc","Deg C"
* LONGNAME "T_mix"
CSAI "AI4",N,N,"T_mix","Deg C"
* LONGNAME "T_dis"
CSAI "AI5",N,N,"T_dis","Deg C"
***** OUTPUTS *****
* LONGNAME "Bypass EXV"
CSAO "OUT1",N,N,"Bypass","%"
* LONGNAME "Condenser Flow"
CSAO "OUT2",N,N,"Cond Out","%"
* LONGNAME "SCR Analog Output"
CSAO "OUT9",N,N,"SCR Out","%"
* LONGNAME "Primary EXV"
CSAO "OUT10",N,N,"Prim Out","%"
* LONGNAME "T_suc Display"
CSAO "OUT11",N,N,"Tsuc Out","%"
* LONGNAME "T_dis Display"
CSAO "OUT12",N,N,"Tdis Out","%"
* LONGNAME "Discharge Output"
CSAO "OUT13",N,N,"Dis Out","%"
CSAO "OUT14",N,N,"AO14","%"
* LONGNAME "Solenoid Valve"
CSBO "DO3",N,N,"Solenoid","OFF","ON"
* LONGNAME "Heat Tape"
CSBO "DO5",N,N,"Heater","OFF","ON"
* LONGNAME "Compressor"
CSBO "DO7",N,N,"Comp","OFF","ON"
***** PROGRAMMABLE MODULES *****
* MODULE LRS1-32 1
CSMODEL "LRS1-32","DX9100"
ADTITLE "Analog Data"
BDTITLE "Binary Data"
CSBD "LRS1",N,N,"Comp","OFF","ON"
CSBD "LRS2",N,N,"Heater","OFF","ON"
CSBD "LRS3",N,N,"Reset","OFF","ON"

* MODULE PID 1
CSMODEL "Bypass","DX9100"
ADTITLE "Analog Data"
BDTITLE "Binary Data"

```

```

CSAD "PM1K1",N,N,"PM1LSP","kPa"
CSAD "PM1K2",N,N,"PM1PB","%"
CSAD "PM1K3",N,N,"PM1TI",""
CSAD "PM1K4",N,N,"PM1TD",""
CSAD "PM1K8",N,N,"PM1EDB","% PB"
CSAD "PM1K11",N,N,"PM1HIL","%"
CSAD "PM1K12",N,N,"PM1LOL","%"
CSAD "PM1K13",N,N,"PM1DHH","kPa"
CSAD "PM1K16",N,N,"PM1DLL","kPa"
CSAD "PM1OU1",N,N,"Byp OCM","%"
CSAD "PM1OU2",N,N,"PM1WSP","kPa"

```

* MODULE PID 2

```

CSMODEL "Discharg","DX9100"
ADTTITLE "Analog Data"
BDTTITLE "Binary Data"
CSAD "PM2K1",N,N,"PM2LSP","kPa"
CSAD "PM2K2",N,N,"PM2PB","%"
CSAD "PM2K3",N,N,"PM2TI",""
CSAD "PM2K4",N,N,"PM2TD",""
CSAD "PM2K8",N,N,"PM2EDB","% PB"
CSAD "PM2K11",N,N,"PM2HIL","%"
CSAD "PM2K12",N,N,"PM2LOL","%"
CSAD "PM2K13",N,N,"PM2DHH","kPa"
CSAD "PM2K16",N,N,"PM2DLL","kPa"
CSAD "PM2OU1",N,N,"Dis OCM","%"
CSAD "PM2OU2",N,N,"PM2WSP","kPa"

```

* MODULE PID 3

```

CSMODEL "SCR PID","DX9100"
ADTTITLE "Analog Data"
BDTTITLE "Binary Data"
CSAD "PM3K1",N,N,"PM3LSP","Deg C"
CSAD "PM3K2",N,N,"PM3PB","%"
CSAD "PM3K3",N,N,"PM3TI",""
CSAD "PM3K4",N,N,"PM3TD",""
CSAD "PM3K8",N,N,"PM3EDB","% PB"
CSAD "PM3K11",N,N,"PM3HIL","%"
CSAD "PM3K12",N,N,"PM3LOL","%"
CSAD "PM3K13",N,N,"PM3DHH","Deg C"
CSAD "PM3K16",N,N,"PM3DLL","Deg C"
CSAD "PM3OU1",N,N,"SCR OCM","%"
CSAD "PM3OU2",N,N,"PM3WSP","Deg C"

```

* MODULE CALC 4

```

CSMODEL "Prim Cal","DX9100"
ADTTITLE "Analog Data"
BDTTITLE "Binary Data"
CSAD "PM4K1",N,N,"PM4K0",""
CSAD "PM4K2",N,N,"PM4K1",""
CSAD "PM4K11",N,N,"PM4HIL","%"
CSAD "PM4K12",N,N,"PM4LOL","%"
CSAD "PM4OU1",N,N,"Prim OCM","%"

```

* MODULE COMPRTR 5

```

CSMODEL "Compare","DX9100"
ADTTITLE "Analog Data"

```



```

BDTITLE "Binary Data"
CSAD "PM5K1",N,N,"HP SP","kPa"
CSAD "PM5K2",N,N,"HP DF","kPa"
CSAD "PM5K3",N,N,"LP SP","kPa"
CSAD "PM5K4",N,N,"LP DF","kPa"
CSAD "PM5K5",N,N,"HT SP","Deg C"
CSAD "PM5K6",N,N,"HT DF","Deg C"
CSAD "PM5K7",N,N,"TSUC SP","Deg C"
CSAD "PM5K8",N,N,"TSUC DF","Deg C"
CSAD "PM5OU1",N,N,"PM5NCM1","kPa"
CSAD "PM5OU2",N,N,"PM5NCM2","kPa"
CSAD "PM5OU3",N,N,"PM5NCM3","Deg C"
CSAD "PM5OU4",N,N,"Prim dif","Deg C"
CSBD "PM5S1",N,N,"Hi Press","OFF","ON"
CSBD "PM5S2",N,N,"Hi Temp","OFF","ON"
CSBD "PM5S3",N,N,"Lo Press","OFF","ON"
CSBD "PM5S4",N,N,"PM5LS4","OFF","ON"

```

* MODULE CALC 6

```

CSMODEL "Byp Calc","DX9100"
ADTITLE "Analog Data"
BDTITLE "Binary Data"
CSAD "PM6K1",N,N,"PM6K0",""
CSAD "PM6K2",N,N,"PM6K1",""
CSAD "PM6K11",N,N,"PM6HIL","kPa"
CSAD "PM6K12",N,N,"PM6LOL","kPa"
CSAD "PM6OU1",N,N,"Byp WSP","kPa"

```

* MODULE TOTAL 7

```

CSMODEL "Totalize","DX9100"
ADTITLE "Analog Data"
BDTITLE "Binary Data"
CSAD "PM7K1",N,N,"PM7FSL1",""
CSAD "PM7K9",N,N,"PM7FTC1","sec"
CSAD "PM7OU1",N,N,"Prim Int","Deg C"
CSBD "PM7S1",N,N,"PM7FSS1","OFF","ON"
CSBD "PM7CT1",N,N,"PM7HLD1","OFF","ON"

```

* MODULE CALC 8

```

CSMODEL "Dis Calc","DX9100"
ADTITLE "Analog Data"
BDTITLE "Binary Data"
CSAD "PM8K1",N,N,"PM8K0",""
CSAD "PM8K2",N,N,"PM8K1",""
CSAD "PM8K11",N,N,"PM8HIL","kPa"
CSAD "PM8K12",N,N,"PM8LOL","kPa"
CSAD "PM8OU1",N,N,"Dis LSP","kPa"

```

* MODULE DCO 1

```

CSMODEL "Dig Cons","DX9100"
ADTITLE "Analog Data"
BDTITLE "Binary Data"
CSBD "DCO1",N,N,"Shutdown","OFF","ON"
CSBD "DCO2",N,N,"Totalize","OFF","ON"
CSBD "DCO3",N,N,"Solenoid","OFF","ON"
CSBD "DCO4",N,N,"SCR","OFF","ON"
CSBD "DCO5",N,N,"Comp","OFF","ON"

```

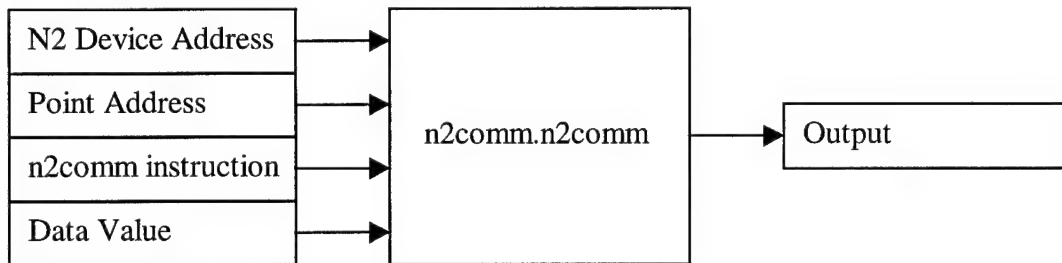
```
* MODULE ACO 1
CSMODEL "A Const","DX9100"
ADTITLE "Analog Data"
BDTITLE "Binary Data"
CSAD "ACO1",N,N,"Byp LSP1","kPa"
CSAD "ACO2",N,N,"Dis LSP1","kPa"
CSAD "ACO3",N,N,"SCR LSP","Deg C"
CSAD "ACO4",N,N,"Prim LSP","Deg C"
CSAD "ACO5",N,N,"Dis LSP2","kPa"
CSAD "ACO6",N,N,"Byp LSP2","kPa"
```

APPENDIX D: VISSIM PROGRAM DESCRIPTION

The Visual Solution's VisSim program is a graphical simulation software tool used to implement an automatic test sequence system. It requires the use of a communications box (see Appendix E for proper connection), the *n2comm.dll* file, and a test point data file *l_stand.map* (printout included in Appendix E). Printouts of the *n2comm.dll* source are included in this Appendix. The user should only be concerned with these source files if there are problems with communications between VisSim and the DX-9100 controller. The most likely cause of problems will occur with the particular COM port that the VisSim program uses for communications. Currently the port is set to COM port 1. In the *talkn2.cpp* file on line 204 of code the port is set as follows: `comPort = 1`. If the COM port being used is different change it to reflect the proper COM port and recompile the *n2comm.dll* file.

If there are further problems check the *error.log* file in the same directory as the *n2comm.dll* file for errors which might have occurred. These errors should provide insight into the cause of the problems.

The *n2comm.dll* file provides a method for writing data to and reading data from the controller. The *n2comm.dll* block in VisSim appears as follows:



The N2 Device Address is the address of the controller being accessed. The Point Address is the address of the point being accessed. Each point in the controller has a unique address. The *n2comm* instruction is one of the following values:

0	Read Analog Value
1	Read Digital Message – 1 Byte
2	Read Digital Message – 2 Byte
3	Write Analog Value
4	Write Digital Message – 2 Byte
5	Write Digital Message – 1 Byte

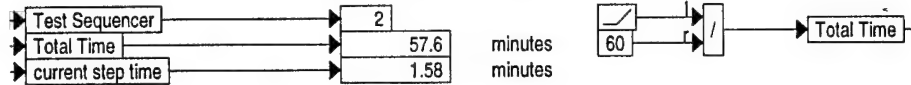
The most frequently used instructions are read analog and write analog. The restriction on the write analog instruction is a 1-bit value or a -127 to 127 range. Because of this limitation the control strategy used involves calculators to achieve the set points desired as discussed in Section 7.2.6 Final Control Flow Diagram. The Data Value is the value to be written to the controller. If a value is being read the Data Value has no effect. The Output is the value read from the controller if a read instruction occurs. Otherwise, during a write instruction, an output of zero means that there was an error in writing the value to the controller and a one means that the instruction completed successfully.

To make programming easier variables can be defined and set by VisSim. This is done to speed up the VisSim simulation process. It takes longer to get a value from the controller than retrieve a variable from the computer memory. By using variables VisSim can retrieve a value from the controller and set it to a variable (such as an Analog Input). Then it does not have to get that value again from the controller the next time it needs that value. Variables are also used for the addresses of the points and program modules to make the program easier to understand. Each specific point, such as local set point, has the same relative address that is added to the base address of the program module being accessed to determine the absolute address. A sample of the *loadstd.vsm* file is included. Because of the size and complexity of the file, the entire file is not included. The printout shows how the program module variable and the point variable are added and input into the *n2comm.n2comm* block for processing instead of using the absolute address for each point. In addition, VisSim has compound blocks that clean up the program. A compound block is just another way of placing a group of blocks on a separate page much like a function or method in conventional programming. Compound blocks can have an input and an output. The shaded blocks in the printout represent compound blocks.

Because the program is so complex a full explanation of the program is not provided here. For an overall view of the program, the *Define Variables* compound block contains all the program code to access the program modules, analog inputs, analog outputs, and digital constants. The digital constants are set using the buttons on the first page of code and a compound block that creates an integer from a binary number. The user can right click these buttons to turn on and off the respective devices. The *Test Sequence Data Import* compound block accesses the *l_stand.map* file to get the set points and analyzes the system data points to

determine when steady state has been reached. When steady state is reached it gets the next test point and writes them to the controller. The *System Input Plots* compound block has plots of the basic system operating points. The first page of code has the important information that the user might be concerned with during simulation. It also has some important variables that should be set for the accuracy desired to determine steady state. The text box above the variables explains what each variable means.

Define Variables
Test Sequence Data Import
System Input Plots



System Measurements

Suction Pressure	534
Suction Temp	7.3906
Discharge Pressure	1148
Discharge Temp	62.094
Primary Valve % Output	3
Bypass Valve % Output	10
SCR % Output	0
Discharge Valve % Output	0
Condenser Valve % Output	60

Set Points

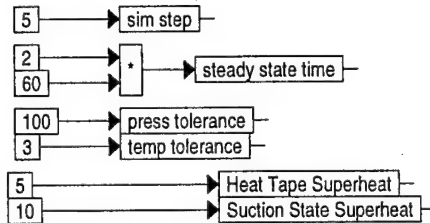
Suction Press Set Point	421
Suction Temp Set Point	5
Discharge Press Set Point	1600
Heat Tape Set Point	0
Heat Tape Correlated Set Point	7.53
Heat Tape Actual Set Point	0
Suction Temp Correlated Set Point	12.5
Suction Temp Actual Set Point	5

System Control Relays

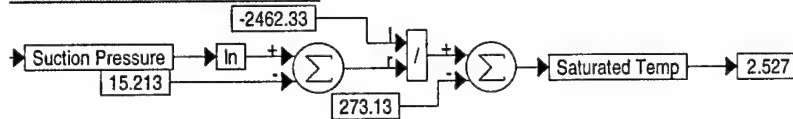
Solenoid Valve Relay	1
SCR Relay	0
Compressor Relay	0
Integral Action Enabled	0

IMPORTANT!

Set "sim step" to simulation step size in simulation setup. Set "steady state time" for amount of time (minutes) to integrate input signals for steady state determination. Set "press tolerance" to tolerance (kPa) desired for pressure steady state and "temp tolerance" to tolerance (deg C) desired for temperature steady state.



R-22 T-P Correlation



Definitions of Variables Used (Numbers represent addresses in controller)

Write Set Points
Analog Input Modules
Analog Output Modules

1 → Controller Address

Program Modules

0	→	General Control Module (#0)
64	→	Bypass Control Module (#1)
160	→	Discharge Pressure Module (#2)
256	→	SCR Control Module (#3)
352	→	Primary Valve OCM Calculator (#4)
448	→	Comparator Module (#5)
544	→	Bypass WSP Calculator (#6)
640	→	Primary Integral Totalizer (#7)
736	→	Discharge WSP Calculator (#8)
1216	→	Analog Input Module #1 (Suction Pressure)
1232	→	Analog Input Module #2 (Discharge Pressure)
1248	→	Analog Input Module #3 (Suction Temperature)
1280	→	Analog Input Module #5 (Discharge Temperature)
1344	→	Analog Output Module #1 (Bypass Valve)
1360	→	Analog Output Module #2 (Condenser Valve)
2304	→	Analog Output Module #9 (Heat Tape)
2320	→	Analog Output Module #10 (Primary Valve)
2368	→	Analog Output Module #13 (Discharge Valve)

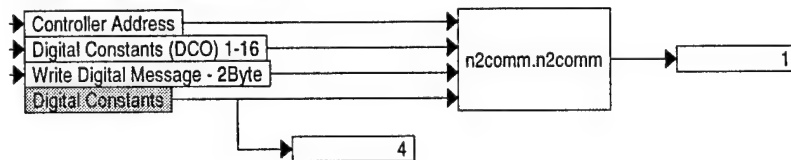
Inputs for N2COMM.DLL

0	→	Read Analog
1	→	Read Digital - 1Byte
2	→	Read Digital - 2Byte
3	→	Write Analog
4	→	Write Digital Message - 2Byte
5	→	Write Digital Message - 1Byte

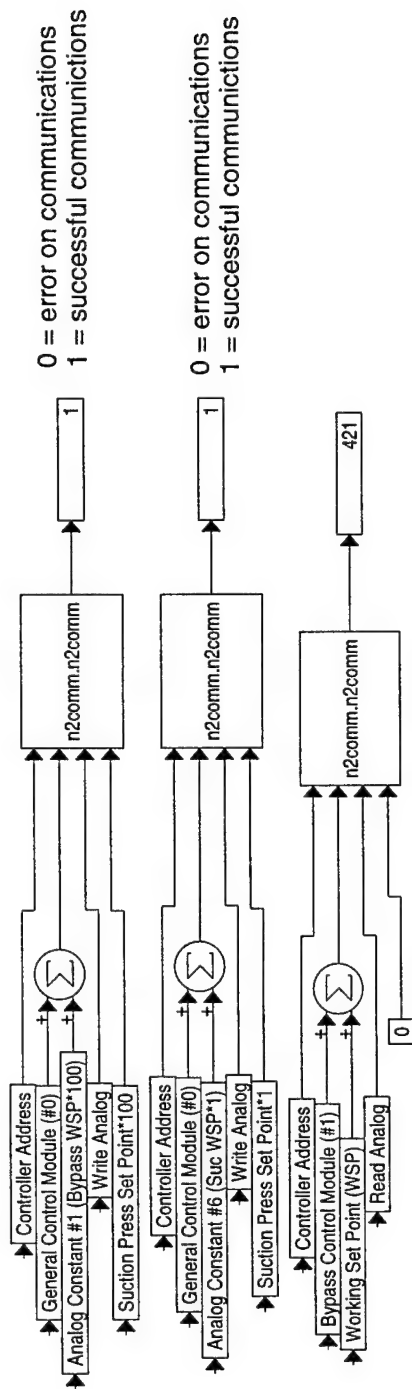
Relative Addresses of Points

10	→	Digital Constants (DCO) 1-16
34	→	Analog Constant #1 (Bypass WSP*100)
35	→	Analog Constant #2 (Dis WSP*100)
36	→	Analog Constant #3 (SCR WSP)
37	→	Analog Constant #4 (Primary WSP)
38	→	Analog Constant #5 (Dis WSP*1)
39	→	Analog Constant #6 (Suc WSP*1)
26	→	Local Set Point (LSP)
27	→	Proportional Band (PB)
61	→	Working Set Point (WSP)
7	→	Analog Input Value
6	→	Analog Output Value

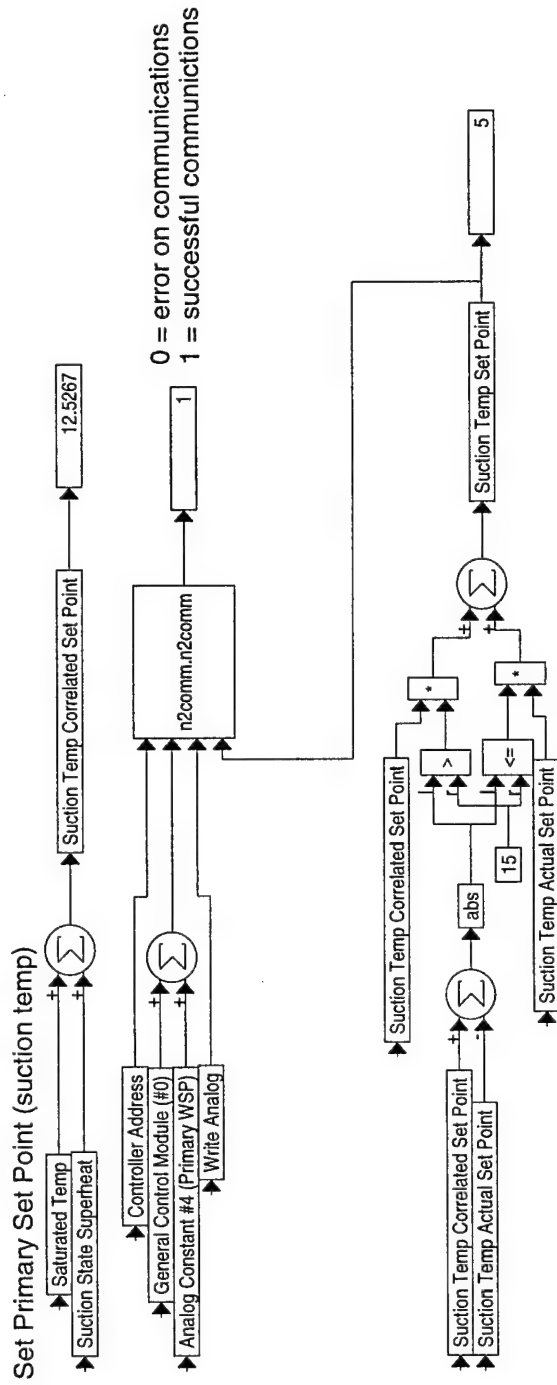
Digital Constants



VisSim-c:\loadst-1\loadstnd.vsm::Define VariablesWrite Set PointsSuction Pressure Set Points

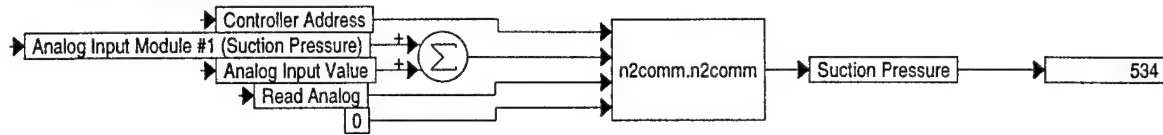


0 = error on communication
1 = successful communication

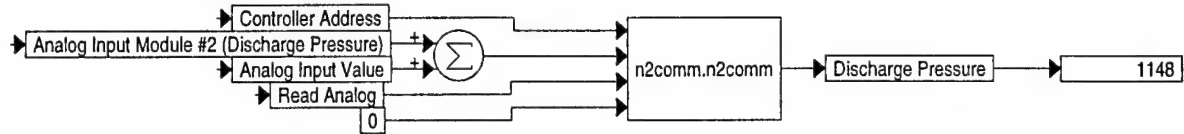


Analog Input Modules

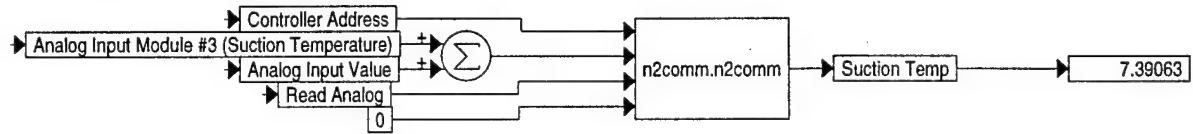
Read Suction Pressure



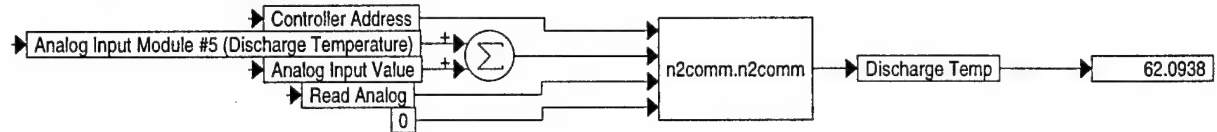
Read Discharge Pressure



Read Suction Temperature

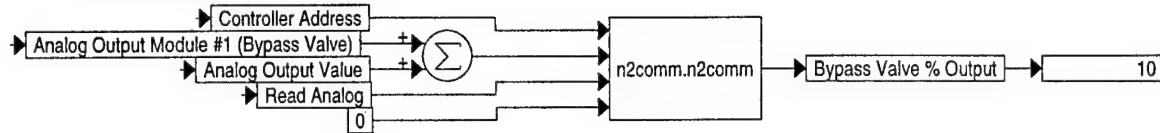


Read Discharge Temperature

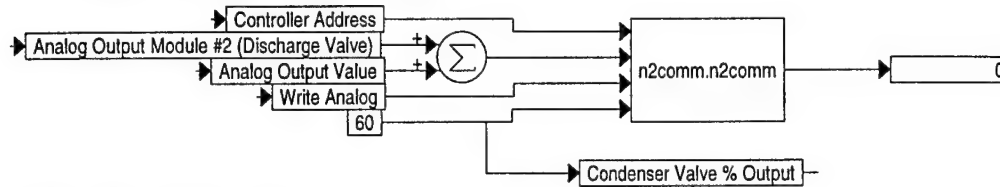


Analog Output Modules

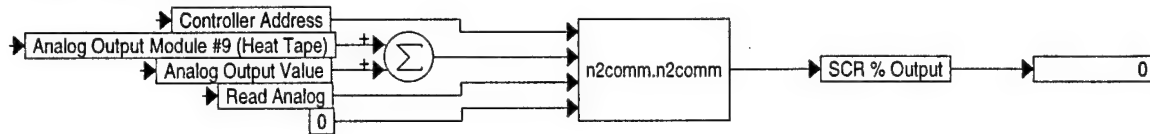
Read Bypass Valve Output



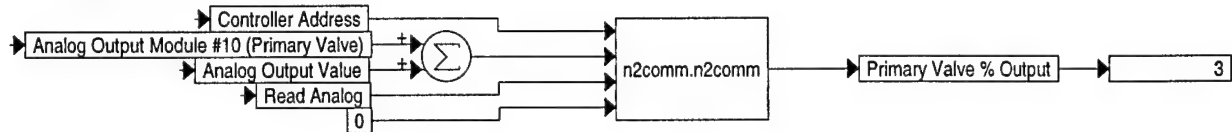
Read Condenser Valve Output



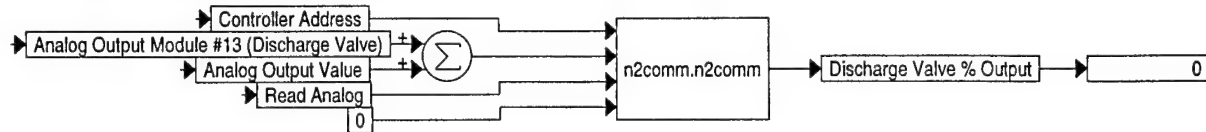
Read Heat Tape Output



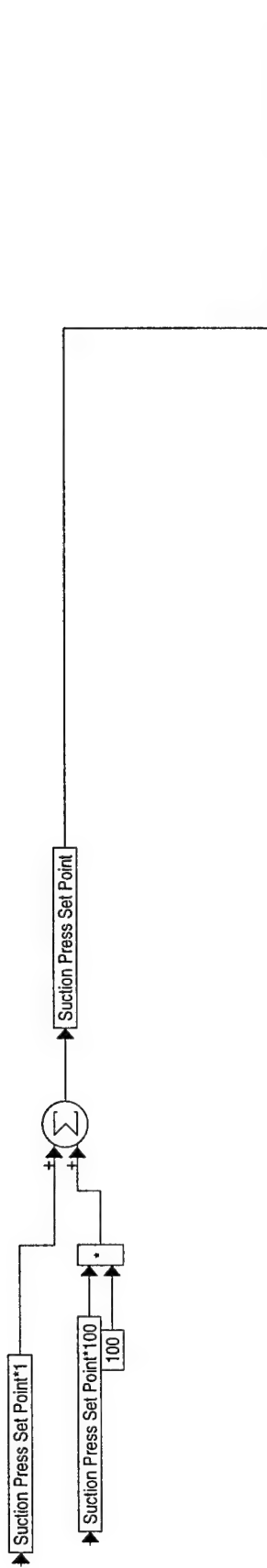
Read Primary Valve Output



Read Discharge Valve Output

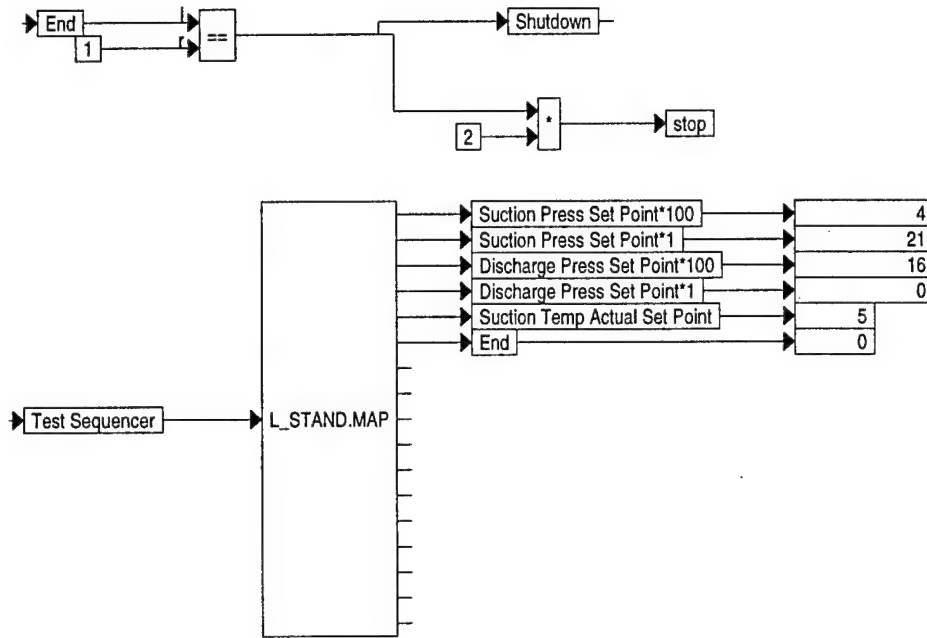


VisSim-c:\loadst-1\loadstnd.vsm::Test Sequence Data ImportSuction Pressure SP



Test Point Data Import

Edit "l_stand.xls" in excel to change test point data. A value of 1 for "End" will cause the test sequence to end.



```
1  #include <math.h>
2  #include <stdio.h>
3  #include <time.h>
4  #include "talkn2.h"
5  #include "errorlog.h"
6
7  extern ErrorLog errorLog;
8
9  char
10 Binary2ASCII(char x)
11 {
12     x%=0x10;
13     if(x<0x0A) {
14         return '0'+x;
15     } else {
16         return 'A'+x-0x0A;
17     }
18 }
19
20 void
21 UChar2ASCII(unsigned char x, char &msb, char &lsb)
22 {
23     msb=Binary2ASCII(x/0x10);
24     lsb=Binary2ASCII(x%0x10);
25 }
26
27 // Given a binary message, this function creates the final ASCII
28 // message ready for the N2 by adding the block check word,
29 // check sum, beginning '>', and ending '\r'.
30 //
31 // message = final ASCII message
32 // B = binary message
33 // nBytes = number of binary bytes in B.
34
35 void
36 GenerateMessage(char* message, unsigned char* B, int nBytes)
37 {
38     int i;
39     unsigned char buf[300];
40     //
41     // buffer the message to make sure there is room for bcc1 and bcc2
42     for(i=0;i<nBytes;i++) buf[i]=B[i];
43     //
44     // generate block check word
45     unsigned short int s1=0,s2=0,bcc1,bcc2;
46     for(i=0;i<nBytes;i++) {
47         s1+=buf[i];
48         s2+=s1;
49     }
50     s1=s1%0x0100+s1/0x0100;
51     s2=s2%0x0100+s2/0x0100;
52     bcc1=s1+s2;
53     bcc1=bcc1%0x0100+bcc1/0x0100;
54     bcc1=(~bcc1)%0x0100;
55     bcc2=s2;
56     buf[nBytes]=bcc1;
57     buf[nBytes+1]=bcc2;
58     //
59     // generate ascii message
60     message[0]='>';
61     for(i=0;i<nBytes+2;i++) {
62         UChar2ASCII(buf[i],message[2*i+1],message[2*i+2]);
63     }
64     //
65     // generate check sum
66     unsigned char checksum=0;
```



```
67     for(i=1;i<2*(nBytes+2)+1;i++) checksum+=(unsigned char)message[i];
68     //
69     // add check sum to end of message
70     UChar2ASCII(checksum,message[2*(nBytes+2)+1],message[2*(nBytes+2)+2]);
71     //
72     // add CR to end of message;
73     message[2*(nBytes+2)+3]='\r';
74     //
75     // NULL terminate
76     message[2*(nBytes+2)+4]=0;
77 }
78
79 void
80 Request_ReadExtendedSingleItem(char * message,int address, unsigned short int
    item)
81 {
82     int i;
83     //
84     // generate binary message
85     unsigned char B[4];
86     B[0]=(unsigned char)address;
87     B[1]=0x84;
88     B[2]=item%0x0100; // LSB
89     B[3]=item/0x0100; // MSB
90     //
91     // generate ascii message
92     GenerateMessage(message,B,4);
93 }
94
95 void
96 Request_WriteExtendedSingleItem_1Byte(char * message,int address, unsigned sh
    ort int item,
97                                     unsigned short int data)
98 {
99     int i;
100    //
101    // generate binary message
102    unsigned char B[5];
103    B[0]=(unsigned char)address;
104    B[1]=0xC4;
105    B[2]=item%0x0100; // LSB
106    B[3]=item/0x0100; // MSB
107    B[4]=data%0x0100; // LSB
108    //
109    // generate ascii message
110    GenerateMessage(message,B,5);
111    return;
112 }
113
114 void
115 Request_WriteExtendedSingleItem_2Byte(char * message,int address, unsigned sh
    ort int item,
116                                     unsigned short int data)
117 {
118     int i;
119     //
120     // generate binary message
121     unsigned char B[6];
122     B[0]=(unsigned char)address;
123     B[1]=0xC4;
124     B[2]=item%0x0100; // LSB
125     B[3]=item/0x0100; // MSB
126     B[4]=data%0x0100; // LSB
127     B[5]=data/0x0100; // MSB
128     //
129     // generate ascii message
```

```

130     GenerateMessage(message,B,6);
131     return;
132 }
133
134 double
135 DX9100Float_Double(unsigned short int x)
136 {
137     // exponent
138     unsigned short int exp = (x&0xF000)>>12;
139     // sign
140     unsigned short int sign = (x&0x0800)>>11;
141     // mantissa
142     unsigned short int mantissa = x&0x07FF;
143     // left-hand-side of decimal point
144     unsigned short int n = 11-exp;
145     double max_rhs = (double)(1<<n);
146     //
147     unsigned short int lhs = mantissa>>n;
148     // right-hand-side of decimal point
149     unsigned short int mask = (1<<n)-1;
150     unsigned short int rhs = mantissa&mask;
151     // value
152     double value = (double)lhs + (double)rhs/max_rhs;
153     if(sign) value *= -1.0;
154     //
155     return value;
156 }
157
158 unsigned short int
159 Double_DX9100Float(double x)
160 {
161     double value;
162     // exponent
163     unsigned short int exp = 7;
164     unsigned short int n = 11-exp;
165     double max_rhs = (double)(1<<n);
166     // sign
167     unsigned short int sign = (x<0) ? 1 : 0;
168     // mantissa
169     value=fabs(x);
170     unsigned short int lhs = (unsigned short int)value;
171     unsigned short int rhs = max_rhs*(value-(double)lhs);
172     unsigned short int mantissa = lhs<<n | rhs;
173     //
174     unsigned short int ret = mantissa |(sign<<11)&0x0800 |(exp<<12)&0xF000;
175     return ret;
176 }
177
178 char*
179 SubString(const char* inStr, int begin, int nBytes)
180 {
181     static char str[200];
182
183     for(int i=0;i<nBytes;i++) {
184         str[i] = inStr[begin+i];
185     }
186     str[nBytes]=0;
187
188     return str;
189 }
190
191 //----- TalkN2 -----
192 TalkN2::TalkN2()
193 {
194     AnalogOutValue=0;
195

```

```
196     int comPort;
197     FILE * fp = fopen("comport.dat","r");
198     if(fp!=NULL) {
199         if(fscanf(fp,"%d",&comPort)!=1) {
200             comPort = 1;
201         }
202         fclose(fp);
203     } else {
204         comPort = 1; //IMPORTANT!!!! set com port here
205     }
206
207     net.Open(comPort,9600);
208 }
209
210 TalkN2::~TalkN2()
211 {
212     net.Close();
213 }
214
215 void
216 TalkN2::ReadAnalogVal()
217 {
218     char buf[100];
219     Request_ReadExtendedSingleItem(buf,address,point);
220     WriteMessage(buf);
221     //
222     char resp[50];
223     ReadResponse(resp,50);
224     //
225     char jnk[100];
226     jnk[0]=resp[3];
227     jnk[1]=resp[4];
228     jnk[2]=resp[1];
229     jnk[3]=resp[2];
230     jnk[4]=0;
231     unsigned short int val;
232     sscanf(jnk,"%x",&val);
233     //
234     AnalogOutValue=DX9100Float_Double(val);
235 }
236
237 void
238 TalkN2::ReadDigitalVal_1Byte()
239 {
240     char buf[100];
241     Request_ReadExtendedSingleItem(buf,address,point);
242     WriteMessage(buf);
243     //
244     char resp[50];
245     ReadResponse(resp,50);
246     //
247     char jnk[100];
248     jnk[0]=resp[1];
249     jnk[1]=resp[2];
250     jnk[2]=0;
251
252     unsigned short int val;
253     sscanf(jnk,"%x",&val);
254     DigitalOutValue = val;
255 }
256
257 void
258 TalkN2::ReadDigitalVal_2Byte()
259 {
260     char buf[100];
261     Request_ReadExtendedSingleItem(buf,address,point);
```

```
262     WriteMessage(buf);
263     //
264     char resp[50];
265     ReadResponse(resp,50);
266     //
267     char jnk[100];
268
269     jnk[0]=resp[3];
270     jnk[1]=resp[4];
271     jnk[2]=resp[1];
272     jnk[3]=resp[2];
273     jnk[4]=0;
274     unsigned short int val;
275     sscanf(jnk,"%x",&val);
276     DigitalOutValue = val;
277 }
278
279 void
280 TalkN2::WriteDigitalMessage_1Byte()
281 {
282     char buf[100];
283     Request_WriteExtendedSingleItem_1Byte(buf,address,point,inValue);
284     WriteMessage(buf);
285     //
286     char resp[50];
287     ReadResponse(resp,50);
288     //
289     if (resp[0]=='A')
290         DigitalOutValue = 1;
291     else
292         DigitalOutValue = 0;
293 }
294
295 void
296 TalkN2::WriteDigitalMessage_2Byte()
297 {
298     char buf[100];
299     Request_WriteExtendedSingleItem_2Byte(buf,address,point,inValue);
300     WriteMessage(buf);
301     //
302     char resp[50];
303     ReadResponse(resp,50);
304     //
305     if (resp[0]=='A')
306         DigitalOutValue = 1;
307     else
308         DigitalOutValue = 0;
309 }
310
311 void
312 TalkN2::WriteAnalogConstant_2Byte()
313 {
314     char buf[100];
315
316     Request_WriteExtendedSingleItem_2Byte(buf,address,point,Double_DX9100Floa
317 t(inValue));
318     WriteMessage(buf);
319     //
320     char resp[50];
321     ReadResponse(resp,50);
322     //
323     if (resp[0]=='A')
324         DigitalOutValue = 1;
325     else
326         DigitalOutValue = 0;
```

```
327
328 void
329 TalkN2::WriteMessage(char* str)
330 {
331     net.Write(str,strlen(str));
332 }
333
334 void
335 TalkN2::ReadResponse(char* buf,int maxlen)
336 {
337     int nBytes=0;
338     time_t t_start = time(NULL);
339     while (time(NULL)<t_start+5) {
340         int bytesRead = net.Read(buf+nBytes,maxlen-nBytes);
341         if (bytesRead>0) {
342             nBytes+=bytesRead;
343             buf[nBytes]=0;
344             if (strchr(buf,'\r')!=NULL) return;
345         }
346     }
347     char str[128];
348     sprintf(str,"Time out reading response. %d bytes read.",nBytes);
349     errorLog.Add("TalkN2::ReadResponse",str);
350 }
351
352
```

```
1  #include <time.h>
2  #include "errorlog.h"
3
4  ErrorLog errorLog;
5
6  ErrorLog::ErrorLog()
7  {
8      ClearError();
9
10     fp = fopen("error.log","w");
11     if(fp) {
12         char da[32],ti[32];
13         _strdate(da);
14         _strtime(ti);
15         fprintf(fp,"Starting error log %s %s.\n\n",da,ti);
16     }
17 }
18
19 ErrorLog::~ErrorLog()
20 {
21     if(fp) fclose(fp);
22 }
23
24 void
25 ErrorLog::ClearError()
26 {
27     isError=0;
28 }
29
30 void
31 ErrorLog::AddWarning(const char* CLASS, const char* METHOD, const char* msg)
32 {
33     //if(fp) fprintf(fp,"Warning! %s::%s >>> %s\n",CLASS,METHOD,msg);
34 }
35
36 void
37 ErrorLog::Add(const char* func, const char* msg)
38 {
39     isError=1;
40     if(fp) fprintf(fp,"Fatal! %s >>> %s\n",func,msg);
41 }
42
```

```
1  #include <stdio.h>
2  #include <time.h>
3  #include <sys\timeb.h>
4  #include "netcomm.h"
5  #include "errorlog.h"
6
7  extern ErrorLog errorLog;
8
9  static void
10 wait(double dt)
11 {
12     timeb t;
13     double t0,t1;
14
15     // get start time
16     ftime(&t);
17     t0=(double)t.time+0.001*(double)t.millitm;
18
19     //wait one second
20     do {
21         ftime(&t);
22         t1=(double)t.time+0.001*(double)t.millitm;
23     } while (t1-t0<dt);
24 }
25
26 /*****
27 NetComm: No argument constructor
28 *****/
29 NetComm::NetComm()
30 {
31     idCOM=-1;           // com port id
32     InQueueSize=512;    // input buffer queue size
33     OutQueueSize=512;   // output buffer queue size
34
35     Port=-1;            // Com port
36     Parity='N';         // Parity ('N','E','O')
37     Baud=9600;          // Baud Rate
38     DataBits=8;         // Data bits
39     StopBits=1;         // Stop bits
40
41     EchoOn=0;
42 }
43
44 /*****
45 NetComm: Open communication channel
46 Argument [com] sets the communication port (1-4).
47 Returns 0 if successful, 1 if com port already active,
48     2 if OpenComm() fails, 3 if BuildCommDCB() fails,
49     and 4 if SetCommState() fails.
50 *****/
51 int
52 NetComm::Open(int com, long baud_rate)
53 {
54     char str[40];
55
56     // Test if commport is already open
57     if(IsOpen()) return 1;
58
59     // set baud rate
60     Baud = baud_rate;
61
62     // Open the communication port
63     sprintf(str,"COM%d",com);
64     idCOM = OpenComm(str,InQueueSize,OutQueueSize);
65     if(idCOM<0) {
66         // Must call GetCommError() to unlock windows.
```

```

67         COMSTAT statCOM;
68         GetCommError(idCOM,&statCOM);
69
70         char str[128];
71         sprintf(str,"Can not open COM %d.  Check COM PORT setting.",com);
72         errorLog.Add("NetComm::Open",str);
73
74         return 2;
75     }
76     Port=com;
77
78     // Set communications port parameters
79     DCB dcb;
80     sprintf(str,"COM%d:%ld,%c,%d,%d",Port,Baud,Parity,DataBits,StopBits);
81     if(BuildCommDCB(str,&dcb)<0) return 3;
82     if(SetCommState(&dcb)<0) return 4;
83
84     return 0;
85 }
86
87 /*****
88 NetComm: String describing com port settings
89 Returns pointer to string describing com port settings.
90 *****/
91 char*
92 NetComm::Settings(void)
93 {
94     static char str[40];
95     sprintf(str,"COM%d:%ld,%c,%d,%d",Port,Baud,Parity,DataBits,StopBits);
96     return str;
97 }
98
99 /*****
100 NetComm: Close communication channel
101 Returns 0 if successful, 1 if not currently open,
102         2 for CloseComm() error
103 *****/
104 int
105 NetComm::Close(void)
106 {
107     if(!IsOpen()) return 1;
108
109     if(CloseComm(idCOM)<0) {
110         // must call to unlock windows
111         COMSTAT statCOM;
112         GetCommError(idCOM,&statCOM);
113         return 2;
114     }
115
116     Port=-1;
117
118     return 0;
119 }
120
121 /*****
122 NetComm: Test if communication channel is open
123 Returns 1 if open, otherwise returns 0.
124 *****/
125 int
126 NetComm::IsOpen(void)
127 {
128     if(Port>=1 && Port<=4) return 1; else return 0;
129 }
130
131 /*****
132 NetComm: Return baud rate

```



```
133  *****/
134  long
135  NetComm::BaudRate(void)
136  {
137      return Baud;
138  }
139
140  /***/
141  NetComm: Write a string over the com port
142  Returns 0 if successful, 1 if com port is not open,
143  2 is write error occurs
144  *****/
145  int
146  NetComm::Write(unsigned char *str, int nBytes)
147  {
148      if(!IsOpen()) return 1;
149
150      EscapeCommFunction(idCOM, SETRTS);
151      //wait(.01);
152      WriteComm(idCOM, str, nBytes);
153      //if(WriteComm(idCOM, str, nBytes)<0) {
154      // // must read error to unlock windoes
155      // COMSTAT statCOM;
156      // GetCommError(idCOM, &statCOM);
157      // return 2;
158      //}
159      wait(.01);
160      EscapeCommFunction(idCOM, CLRRTS);
161
162      return 0;
163  }
164
165
166
167  /***/
168  NetComm: Read a string from the com port
169  Returns number of bytes read if successful, -1 if com
170  port is not open, -2 is write error occurs
171  *****/
172  int
173  NetComm::Read(unsigned char *str, int nBytes)
174  {
175      int ret;
176
177      if(!IsOpen()) return -1;
178
179      if((ret=ReadComm(idCOM, str, nBytes))<0) {
180          // must read error to unlock windoes
181          COMSTAT statCOM;
182          GetCommError(idCOM, &statCOM);
183          return -2;
184      }
185
186      return ret;
187  }
188
189  /***/
190  NetComm: Communicate a message.
191  Communicate synchronizes with the remote unit with some
192  handshaking, sends the message, and returns with the
193  response in [str] if successful. [nBytes] is the
194  number of bytes to send and [maxlen] is the maximum
195  length of the return message. [nRetry] is the
196  number of retries waiting for synchronization.
197  Returns 0 if successful, 1 if not sync, 2 write error,
198  3 no response from message.
```

```

199  *****/
200  int
201  NetComm::Communicate(unsigned char *str, int nBytes, int maxlen, int nRetry)
202  {
203      int n(0);
204      unsigned char Sync[110];
205
206      // synchronize
207      do {
208          if(++n>nRetry) return 1;
209          if(Write(":",1)) return 2;
210      } while (Listen(Sync,100,">",5,0)<0);
211
212      // intercept message if echo on and send echo command
213      if(EchoOn) {
214          str[2] = 255;
215          str[3] = 4;
216      }
217
218      // send message
219      if(Write(str,nBytes)) return 2;
220
221      // listen for response
222      if(Listen(str,maxlen,"\r",30,1)<0) return 3;
223
224      return 0;
225  }
226
227  /*****
228  NetComm: Listen for a response
229  [str] is the character string containing the response,
230  [maxlen] is the maximum length of the response, [sTerm]
231  is the terminating string (e.g., "\r" or "OK"), and
232  [nSec] is the number of seconds before timeout. All
233  strings are null terminated.
234  Returns number of bytes received if successful, -1 if com
235  port is not open, -2 if Read() error, -3 if timeout.
236  *****/
237  int
238  NetComm::Listen(unsigned char *str, int maxlen,
239                  unsigned char *sTerm, int nSec, int LenTest)
240  {
241      time_t t0;
242      int nBytes,nOff(0);
243
244      if(!IsOpen()) {
245          str[0]=0;
246          return -1;
247      }
248
249      time(&t0);
250      while ((int)(time(NULL)-t0)<nSec) {
251          nBytes=Read(str+nOff,maxlen-nOff);
252          if(nBytes<0) {
253              str[nOff]=0;
254              return -2;
255          }
256          if(nBytes>0) {
257              nOff+=nBytes;
258              str[nOff]=0;
259              // return if target string found
260              if(strstr((char*)str,(char*)sTerm)!=NULL) {
261                  return nOff;
262              }
263              // return if entire message received
264              //if(LenTest && nOff>0 && nOff>=(int)str[0]) {

```

```
265         // return nOff;
266         //}
267     }
268 }
269
270     return -3;
271 }
272
273 int
274 NetComm::SetEcho(int status)
275 {
276     if(status) {
277         EchoOn = 1;
278     } else {
279         EchoOn = 0;
280     }
281     return EchoOn;
282 }
283
```

```
1  #include "c2cpp.h"
2  #include "talkn2.h"
3
4  static TalkN2 talkN2;
5
6  void
7  SimulationStart(double t)
8  {
9  }
10
11 void
12 SimulationStep(double simTime, double simTimeStep, double * params, double * insi
13 g, double * outsig)
14 {
15     talkN2.Address((int)insig[0]);
16     talkN2.Point((unsigned short int)insig[1]);
17     talkN2.Action((int)insig[2]);
18     talkN2.InValue(insig[3]);
19     if (talkN2.GetAction() == 0) {
20         talkN2.ReadAnalogVal();
21         outsig[0] = talkN2.AOutValue();
22     }
23     else if (talkN2.GetAction() == 1) {
24         talkN2.ReadDigitalVal_1Byte();
25         outsig[0] = talkN2.DOutValue();
26     }
27     else if (talkN2.GetAction() == 2) {
28         talkN2.ReadDigitalVal_2Byte();
29         outsig[0] = talkN2.DOutValue();
30     }
31     else if (talkN2.GetAction() == 3) {
32         talkN2.WriteAnalogConstant_2Byte();
33         outsig[0] = talkN2.DOutValue();
34     }
35     else if (talkN2.GetAction() == 4) {
36         talkN2.WriteDigitalMessage_2Byte();
37         outsig[0] = talkN2.DOutValue();
38     }
39     else if (talkN2.GetAction() == 5) {
40         talkN2.WriteDigitalMessage_1Byte();
41         outsig[0] = talkN2.DOutValue();
42     }
43     else
44         return ;
45 }
46 void
47 SimulationEnd()
48 {
49 }
```

```
1  /* Sampldll.c - Example file for VisSim DLL */
2
3  #include <windows.h> // standard windows include file
4  #include <string.h>
5  #include <stdlib.h>
6  #include "vsuser.h" // vissim function prototypes
7  #include "n2comm.h" // n2comm type definitions and literals
8
9  #include "c2cpp.h"
10
11 /* User menu Table */
12 USER_MENU_ITEM um[] = {
13     /* level 1 menu name, DLL name, -1,-1 */
14     {"Fuzzy", "n2comm", -1,-1}, /* level 1 block menu item */
15     /* menu name, function name, input count, output count */
16     {"fuzzyFun", "n2comm", 5,1, sizeof (FUZZY_INFO), "Fuzzy Rule Interpreter"},
17     /* level 2 menu item #1 */
18     {0} /* End of table */
19 };
20
21 /* This function will be called by VisSim if the DLL in which it resides
22    * is placed in the \windows\visim.ini file with the line:
23    * addon=<dll file name>
24    */
25 int _export PASCAL vsmInit()
26 {
27     /* Each following line adds a block item list to the Blocks menu */
28     setUserBlockMenu(um);
29 }
30
31 char *floatToStr( double d )
32 {
33     static char floatBuf[64];
34     static int sign, dec;
35     char *p = gcvt( d, 10, floatBuf);
36     /* skip leading minus */
37     if (*p == '-') p++;
38     /* remove leading space */
39     while (*p == ' ') strcpy (p, p+1);
40     /* Remove leading zeros, (be sure number has a digit left) */
41     if (*p == '0' && p[1]) strcpy (p, p+1);
42     /* Remove trailing '.' for 1. and 1.e12 */
43     p = floatBuf;
44     while (*p && *p != '.') p++;
45     if (*p == '.')
46     { if (p[1] == 0) *p = 0;
47       else if (tolower(p[1]) == 'e') strcpy (p, p+1);
48     }
49     /* set to zero if removed everything */
50     if (!*floatBuf) strcpy( floatBuf, "0");
51     return floatBuf;
52 }
53
54 void setDlgItemFlt( HWND hDlg, int id, double val)
55 {
56     SetDlgItemText( hDlg, id, floatToStr(val));
57 }
58
59 double getDlgItemFlt( HWND hDlg, int id)
60 {
61     static char valBuf[80];
62     GetDlgItemText( hDlg, id, valBuf, sizeof (valBuf));
63     return atof(valBuf);
64 }
65
```

```
66  /* globals */
67
68  HINSTANCE DLLInst;
69
70
71  /* simulation start function */
72
73  void FAR EXPORT PASCAL n2commSS(params, runCount)
74  double FAR params[];
75  long FAR *runCount;
76  {
77  /* Do simulation startup sorts of things here */
78      double simTime;
79      getSimTime(&simTime);
80      //
81      SimulationStart(simTime);
82  }
83
84
85  /* simulation step function */
86  void FAR EXPORT PASCAL n2comm(params, insig, outsig)
87  double FAR params[];
88  double FAR insig[];
89  double FAR outsig[];
90  {
91      /* Do simulation time step things here */
92      double simTimeStep;
93      double simTime;
94      getSimTimeStep(&simTimeStep);
95      getSimTime(&simTime);
96
97      SimulationStep(simTime, simTimeStep, params, insig, outsig);
98  }
99
100
101  /* simulation end function */
102  void FAR EXPORT PASCAL n2commSE(double FAR params[], long FAR*runCount)
103  {
104      /* Do simulation end time things here */
105
106      SimulationEnd();
107  }
108
109
110  /* Create and display custom dialog */
111  static int showCustomDialog(values)
112  FUZZY_INFO FAR *values;
113  {
114      return DialogBoxParam(DLLInst, "SAMPLDLL", NULL, N2CommProc, (LPARAM) values);
115  }
116  /* n2commPC */
117
118
119  #define FUZZY_INFO_STR "i[4]c[%d]"
120  static char fmtBuf[128];
121
122  /* user Event function to set block name and handle parameter discription f
or auto file save/restore */
123  LPSTR PASCAL EXPORT n2commEvent(HWND h, int msg, WPARAM wParam, LPARAM lParam)
124  {
125      FUZZY_INFO FAR*fzParam;
126      LPARAM blockHandle;
127      long arg;
128      SIM_INFO simInfo;
```

```
129     OPT_INFO optInfo;
130
131     switch (msg) {
132
133         case WM_VSM_GET_PARAM_DESC:
134             wsprintf( fmtBuf, FUZZY_INFO_STR, MAX_BLOCK_NAME); /* Use sprintf so f
135             ield size gets set */
136             return fmtBuf;
137
138         case WM_VSM_GET_BLOCK_NAME: /* returns block name */
139             fzParam = (FUZZY_INFO FAR*)lParam;
140             return fzParam->name[0]?fzParam->name:0; // If no name, return null
141
142         case WM_VSM_BLOCK_SETUP: /* User clicked right mouse button */
143             blockHandle = lParam;
144             fzParam = (FUZZY_INFO FAR*)vissimRequest(VR_GET_BLOCK_PARAMS, blockHan
145             dle, 0); // Get params from block handle
146             vissimRequest(VR_GET_VISSIM_STATE, &simInfo, sizeof(simInfo));
147             vissimRequest(VR_GET_GLOBAL_OPT_INFO, &optInfo, sizeof(optInfo));
148             if (IDOK == showCustomDialog(fzParam))
149             { // If user said OK, set block connector counts
150                 arg = fzParam->inputCount | (((long)fzParam->outputCount)<<16);
151                 vissimRequest(VR_SET_BLOCK_CONNECTOR_COUNT, blockHandle, arg);
152             }
153             return 0;
154
155         case WM_VSM_ADD_CONNECTOR: // These messages sent when user tries to add
156         or del connectors on block
157         case WM_VSM_DEL_CONNECTOR:
158             return 0; // return of 0 says don't allow users to change connect co
159     unt
160     }
161     return 0;
162 }
163
164 /* parameter allocation function */
165 long FAR EXPORT PASCAL n2commPA(pCount)
166 short FAR *pCount;
167 {
168     *pCount = 0;
169     return (sizeof(FUZZY_INFO));
170 }
171
172 /* parameter initialization function */
173 void FAR EXPORT PASCAL n2commPI(values)
174 FUZZY_INFO FAR *values;
175 {
176     values->iInfMethod = INF_MAX_MIN;
177     values->iDefuzMethod = DEFUZ_YAGERS;
178     values->inputCount = 3;
179     values->outputCount = 2;
180 } /* n2commPI */
181
182 /* parameter change dialog procedure */
183
184 BOOL FAR EXPORT PASCAL N2CommProc(hDlg, msg, wParam, lParam)
185 HWND hDlg;
186 UINT msg;
187 WPARAM wParam;
188 LPARAM lParam;
189 {
190     static FUZZY_INFO FAR *data;
191     int iInference, iDefuz;
```

```
191
192     switch (msg) {
193
194     case WM_INITDIALOG:
195         /* initialize the dialog box fields */
196         data = (FUZZY_INFO FAR *) lParam;
197         iInference = IDD_FIRST_INFERENCE + data->iInfMethod;
198         iDefuz = IDD_FIRST_DEFUZ + data->iDefuzMethod;
199         CheckRadioButton(hDlg, IDD_MINMAX, IDD_BOUNDSUMPROD, iInference);
200         CheckRadioButton(hDlg, IDD_YAGER, IDD_MAXHEIGHT, iDefuz);
201         SetDlgItemText( hDlg, IDD_NAME, data->name);
202         break;
203
204     case WM_COMMAND:
205         switch (wParam) {
206         case IDCANCEL:
207             EndDialog(hDlg, FALSE);
208             break;
209
210         case IDOK:
211             /* read the new values */
212             if (IsDlgButtonChecked(hDlg, IDD_MINMAX))
213                 data->iInfMethod = INF_MAX_MIN;
214             else if (IsDlgButtonChecked(hDlg, IDD_SUMPROD))
215                 data->iInfMethod = INF_SUM_PRODUCT;
216             else
217                 data->iInfMethod = INF_BOUNDED_SUM;
218
219             if (IsDlgButtonChecked(hDlg, IDD_YAGER))
220                 data->iDefuzMethod = DEFUZ_YAGERS;
221             else if (IsDlgButtonChecked(hDlg, IDD_CENTER))
222                 data->iDefuzMethod = DEFUZ_CENTER_GRAVITY;
223             else
224                 data->iDefuzMethod = DEFUZ_MAX_HEIGHT;
225
226             GetDlgItemText( hDlg, IDD_NAME, data->name, sizeof (data->name));
227             EndDialog(hDlg, TRUE);
228             break;
229         }
230         return TRUE;
231     }
232     return FALSE;
233 } /* SampleDllProc */
234
235
236
237 /* DLL entry point */
238
239 int FAR EXPORT PASCAL LibMain(hInstance, wDataSeg, cbHeapSize, lpszCmdLine)
240 HINSTANCE hInstance;
241 WORD wDataSeg;
242 WORD cbHeapSize;
243 LPSTR lpszCmdLine;
244 {
245     #ifdef UNIX
246         LibFunc("n2comm", n2comm);
247         LibFunc("n2commPA", n2commPA);
248         LibFunc("n2commPI", n2commPI);
249         LibFunc("n2commPC", n2commPC);
250         LibFunc("n2commSE", n2commSE);
251         LibFunc("n2commSS", n2commSS);
252     #endif
253     DLLInst = hInstance;
254     return TRUE;
255 } /* LibMain */
256
```



```
257
258  /* DLL exit procedure */
259  int FAR PASCAL WEP(nParameter)
260  int nParameter;
261  {
262      return 1;
263  } /* WEP */
264
```

APPENDIX E: OPERATING INSTRUCTIONS

This Appendix is meant to be used as an operations manual. It will provide the guidance necessary to properly and safely operate the compressor load stand. For safety concerns, the user must obtain a safety check on the load stand before operating it unsupervised. This appendix should provide the user with the knowledge required to pass the safety check and become efficient in the operation of the equipment. It is suggested that the user operate the equipment under supervision to become familiar with the software and the operating characteristics of the load stand before taking the safety check and attempting to operate it unsupervised.

E.1 Hardware Setup

The load stand must be connected to a 240V/3 ϕ power supply. The load stand also requires water for the condenser. Make sure the water supply hose is connected to a supply fixture and the return hose is routed to a drain. Turn on cold water to the condenser. Confirm that condenser water shut-off valves are open (handle parallel to the floor). Turn the power supply to the load stand on. Turn the load stand "SYSTEM" power switch on and confirm all voltage lights are on and that equilibrium pressures and temperatures are reading correctly (approximately 850 kPa and 25 °C for R-22) on the displays. Make sure that the cooling fan for the compressor is turned on and blowing air across the compressor. Turn on the "COMP" power switch.

To program or commission the controller a cable with the Johnson Controls Metasys adapter on one end must be connected to a COM port and the other end to the N2 bus port on the controller. The Metasys adapter needs to be plugged in to an outlet.

In order to take data, a Hewlett Packard data acquisition unit must be used as well as a computer with an interface card. Ensure that the load stand data acquisition card's connectors are properly connected to the quick disconnects on the side of the load stand cabinet according to the following table:

Load Stand Quick Connect Number	HP Card Quick Connect Number	Sensor
T1	W6	Suction Pressure
T2	W7	Discharge Pressure
T3	W8	Condensing Pressure

T5	W9	Expansion Pressure
T6	W10	Mixing Pressure
T7	W11	Mass Flow Meter
T8	W12	Watt Transducer
B1	Y1	Discharge Temperature
B2	Y2	Condensing Temperature
B3	Y3	Expansion Temperature
B5	Y4	Mixing Temperature
B6	Y0	Suction Temperature

“T” refers to the top panel of quick connects on the load stand. Similarly “B” refers to the bottom panel. “W” refers to the white quick connects and “Y” to the yellow. Insert the load stand data acquisition card into the second slot of the HP data acquisition unit. Make sure the data acquisition unit is connected to the computer.

For the automatic test sequence system, the communications box should be hooked up as follows:

Cable #1: connect to the N2 bus port on the DX-9100 controller

Cable #2: connect to 24 VAC power supply (currently provided by the load stand electrical system)

Cable #3: connect to COM port 1 of the controlling computer

Cable #4: connect to an available COM port of the controlling computer

Note that Cable #4 should have a Johnson Controls Metasys adapter. Cable #4 is only needed to use the Johnson Controls Programming Tool or Point Template to communicate with the controller.

E.2 Software Setup

To program or commission the controller the computer should have the Johnson Controls GX-9100 Programming Tool Ver 3.0 and Johnson Controls Point Template Ver 5.10 or newer software installed. To change the control strategy in the controller start the Programming Tool and open the *loadstnd.dxs* file. Make the appropriate changes and save it. To download the new

strategy into the controller make sure the controller has power and that neither the Point Template nor Visual Solution's VisSim programs are communicating with the controller. Choose the *Actions → Download* menu item. In the dialog box, under 'Item' check DX, under 'Port' check the appropriate COM port, and set the 'Address' to the appropriate address set by the DIP switches on the bottom of the controller (load stand controller set to 1 currently). After downloading is complete the program can be closed.

To use the commissioning tool start the Point Template software. Open the *load.dmo* file (printout included in Appendix C). Check for other programs communicating with the controller. Choose the *commission→ current configuration* menu item. Check the appropriate COM port, check the N2 bus radio button, and set the N2 device address to the appropriate address. The program has three different screens: inputs, outputs, and parameters. Click on the buttons to see the different screens. By monitoring the outputs screens you can see what the control signals are doing as the system operates. To change an output to a constant value double click it and enter the desired value. Then click the override button. That control output is held at that value and the feedback control loop is being overridden. Overridden outputs will have identifying marks beside them. To release an overridden output, double click it and choose the release button. To change an overridden output just double click it and enter the new value and choose override. To change parameters such as the set points, gains, or digital constants use the same method for outputs. To test the communications change the condenser output to 100% and check to see if the water flow increases. Release the condenser output. The Point Template is an extremely effective tool for tuning the controller. By changing the control values you can see the effect immediately without having to download a new strategy. When finished using the commissioning tool choose the *Commission→Exit commissioning mode* menu item and then the *File→Exit* menu item. Note: when you exit all overrides will be released.

To use the data acquisition software developed the computer must have the HP Vee Version 3.12 or newer software installed on it. Turn on the power to the HP unit and open the *loadstdn.vee* file in HP Vee. In the HP Vee program, make sure the Write Data checkbox is checked. Adjust the Write Freq and Disp Freq to the desired interval in seconds for displaying and writing data. Note: the interval for displaying data must be less than or equal to the writing frequency. Click on the Start button in HP Vee. Give a file name for the data file and a brief

description of the test being conducted. A sample data file is included. The graphs plot the data points at the interval entered in Disp Freq. To stop the data acquisition, click the Stop button at the top of the screen. To access the data recorded, open the data file using any word processor and copy the data into a spreadsheet for analysis. A sample printout of a data file is included to show how the format in which the data is recorded.

To use the automatic test sequence system Visual Solution's VisSim Ver 2.0 program must be installed on the computer and the communications box must be used. VisSim cannot be running at any time while other programs are trying to communicate with the controller. Ensure that the *l_stand.map* file (printout included) is present in the same directory as the *loadstand.vsm* file. To program new test points, open the *l_stand.xls* file in Excel (printout included). Change the set points to reflect the desired test points. Make sure that the first column starts at one for the first test point and increments by one for each test point. Also ensure that the end variable is zero except for a repeat of the last test point. The end variable initializes the shut down in VisSim so make an extra test point after the last desired test point. After all test points have been entered copy only the test points (none of the text at the top of the file) into a new Excel file. Save this file as a tab-delimited text file called *l_stand.map*. Now VisSim can be started and *loadstnd.vsm* can be opened. To run a test sequence choose the *Simulation → Go* menu item or the green arrow on the toolbar. This will start the simulation. Now start the compressor and VisSim will proceed through the test points.

If problems are encountered with VisSim, take the cover off of the communications box and make sure that the power switch is turned on (green light should be lit). Check to make sure that when VisSim is trying to communicate with the controller two amber lights on the card in the box blink rapidly. If the amber lights are not blinking the COM port set in the *n2comm.dll* source code might be different than the one being used. Either change the code as described in Appendix D or switch Cable #3 to the other COM port. Changing the COM port is the easiest method if both COM ports are available. This should correct the problem. If not, check the VisSim simulation program for errors.

E.3 Checklists

The following checklists are provided to aid in system start-up, operation, and shut-down.

E.3.1 Pre-Operation Checklist

1. Become familiar with laboratory surroundings including locations and types of fire extinguishers, locations of exits, and location of telephone and emergency phone numbers.
2. Check water hoses for proper connection (supply into condenser and return into drain).
3. Open condenser water shut-off valves.
4. Connect data acquisition unit to load stand quick connects.
5. Connect communications box to controller, power supply, and computer COM ports.
6. Set up compressor cooling fan and turn on.
7. Inspect load stand piping and electrical wiring for any abnormalities.

E.3.2 Operation Checklist

1. Turn cold water supply on.
2. Turn main power supply on. Leave extension arm in bus bar disconnect for emergency shutdown.
3. Turn computer on. Edit *l_stand.map* for proper test points.
4. Turn data acquisition unit on.
5. Turn "SYSTEM" power switch on.
6. Turn "COMP" power switch on.
7. Check voltage lights and displays for proper indication.
8. Check compressor inverter for proper frequency.
9. Visually inspect condenser water outlet for water flow.
10. Start HP Vee and open *loadstnd.vee*.
11. Start VisSim and open *loadstnd.vsm*.
12. Adjust operating variables in HP Vee (Write Data, Write Freq, Disp Freq) and click the Start button.
13. Provide a filename and description for experiment.
14. Verify that HP Vee is working properly (graphs show data points).
15. Start VisSim simulation.
16. Push Run button on compressor inverter.
17. Monitor system for proper operation.

18. Turn "SYSTEM" switch off if problems occur.

E.3.3 Post-Operation Checklist

1. Push Stop button on compressor inverter if compressor is still running.
2. Stop VisSim simulation and close program.
3. Stop HP Vee and close program.
4. Verify that the data file was created. Copy data from data file into a spreadsheet for analysis.
5. Open Programming Tool and *loadstnd.dxs* file.
6. Download default control strategy into controller.
7. Close Programming Tool.
8. Shut down computer.
9. Turn data acquisition unit off.
10. Turn "COMP" switch off.
11. Turn "SYSTEM" switch off.
12. Turn main power supply off and remove extension arm from disconnect. Verify that power is disconnected by turning "SYSTEM" switch on and verifying that power is not on.
13. Turn water supply off. Verify that water flow is turned off.
14. Turn compressor cooling fan off.

To create MAP file, copy data (no headers) to new sheet & save as text (tab delimited)

Step	Suction Press Set Point * 100	Suction Press Set Point * 1	Discharge Press Set Point * 100	Discharge Press Set Point * 1	Suction Temp Set Point	End	
1.00	3.00	54.00	16.00	0.00	0.00	0	
2.00	4.00	21.00	16.00	0.00	5.00	0	
3.00	4.00	21.00	16.00	0.00	5.00	0	
4.00	4.00	97.00	17.00	28.00	0.00	0	
5.00	4.00	97.00	13.00	54.00	0.00	0	
6.00	4.00	97.00	10.00	44.00	0.00	0	
7.00	6.00	80.00	17.00	28.00	10.00	0	
8.00	6.00	80.00	13.00	54.00	10.00	0	
9.00	6.00	80.00	10.00	44.00	10.00	0	
10.00	6.00	80.00	10.00	44.00	10.00	1	

l_stand.map

1	3	54	16	0	0	0
2	4	21	16	0	5	0
3	4	21	16	0	5	0
4	4	97	17	28	0	0
5	4	97	13	54	0	0
6	4	97	10	44	0	0
7	6	80	17	28	10	0
8	6	80	13	54	10	0
9	6	80	10	44	10	0
10	6	80	10	44	10	1

Sample Test Data File

Time of Experiment: Fri 10/Apr/1998 12:59

Summary of Experiments: PI controllers Test point #1: 421/1728 - no SCR

Time in minutes, temperatures in degrees Celsius, pressures in kPa, and mass flow in kg/hr.

Time	T_1	T_2	T_3	T_4	T_5	Time	P_1	P_2	P_3	P_4	P_5	Time	m_dot	Errors
0.03	+11.94	+59.68	+19.39	+17.08	+27.58	0.03	759.16	782.08	783.81	761.97	758.84	0.03	0.434	O
0.12	+12.70	+59.63	+20.10	+17.28	+29.32	0.12	770.37	810.24	799.90	766.90	774.22	0.12	0.884	O
0.20	+16.04	+75.34	+20.17	+17.16	+17.89	0.20	751.03	1103.63	823.95	746.84	746.54	0.20	22.785	O
0.28	+10.88	+78.54	+19.98	+16.42	+12.71	0.28	593.30	1567.22	858.34	590.71	602.18	0.28	39.118	O
0.37	+2.60	+78.91	+22.37	+15.96	+4.85	0.37	438.27	1939.27	943.20	424.27	446.61	0.37	46.940	O
0.45	+1.63	+77.20	+22.98	+14.55	+3.64	0.45	327.73	1683.12	984.49	311.47	339.71	0.45	47.667	O
0.53	+4.20	+78.25	+22.75	+13.95	-0.78	0.53	376.87	1660.27	955.14	366.45	395.19	0.53	50.488	O
0.62	+4.90	+77.90	+22.42	+14.06	-1.99	0.62	368.87	1643.36	945.88	357.60	386.84	0.62	58.982	O
0.70	+5.94	+78.54	+22.65	+15.06	-1.22	0.70	373.40	1650.59	932.80	363.72	393.26	0.70	62.468	O
0.78	+5.89	+78.33	+22.12	+15.47	-1.10	0.78	385.38	1677.78	920.75	374.14	406.95	0.78	59.943	O
0.87	+6.65	+78.50	+22.12	+16.39	-0.57	0.87	383.84	1684.09	914.68	370.99	403.63	0.87	56.647	O
0.95	+7.17	+78.59	+22.12	+17.05	+0.31	0.95	376.74	1670.50	908.29	365.16	399.84	0.95	53.985	O
1.03	+7.89	+79.17	+22.12	+18.03	+2.49	1.03	375.20	1648.98	889.86	365.26	401.01	1.03	54.438	O
1.12	+8.58	+79.57	+22.26	+18.98	+4.15	1.12	379.45	1642.81	878.82	367.14	403.74	1.12	51.325	O
1.20	+9.11	+79.69	+22.01	+19.64	+5.63	1.20	385.99	1701.53	870.07	374.25	411.78	1.20	53.162	O
1.28	+9.86	+80.11	+22.16	+20.62	+7.31	1.28	391.95	1654.26	863.12	380.16	417.63	1.28	52.242	O
1.37	+10.45	+80.44	+22.05	+21.33	+8.67	1.37	399.79	1672.28	859.21	387.30	421.92	1.37	52.197	O
1.45	+11.20	+80.90	+22.14	+22.15	+10.16	1.45	402.95	1689.18	858.04	392.35	429.92	1.45	52.235	O
1.53	+11.87	+81.21	+22.18	+22.76	+11.60	1.53	408.37	1701.53	854.55	397.44	433.36	1.53	52.273	O
1.62	+11.53	+81.21	+21.98	+23.28	+12.59	1.62	411.43	1714.04	854.24	400.48	435.94	1.62	52.830	O
1.70	+13.39	+81.94	+22.34	+24.05	+14.18	1.70	416.03	1732.53	854.33	404.54	440.60	1.70	52.909	O
1.78	+13.96	+82.28	+22.36	+24.58	+15.29	1.78	418.08	1732.85	855.70	405.86	439.34	1.78	53.179	O
1.87	+14.57	+82.46	+22.27	+24.96	+16.23	1.87	417.87	1741.71	856.62	408.05	443.17	1.87	53.985	O
1.95	+15.12	+82.64	+22.32	+25.30	+17.18	1.95	419.62	1748.73	857.02	409.07	443.55	1.95	53.798	O
2.03	+15.74	+83.08	+22.28	+25.65	+18.05	2.03	421.72	1745.42	858.91	410.11	444.19	2.03	52.484	O
2.12	+16.38	+83.33	+22.46	+26.08	+19.04	2.12	421.16	1752.70	859.70	411.97	445.21	2.12	51.360	O
2.20	+16.75	+83.33	+22.27	+26.17	+19.60	2.20	424.12	1756.06	859.28	412.70	443.05	2.20	53.155	O
2.28	+17.45	+83.84	+22.46	+26.52	+20.51	2.28	425.40	1756.41	859.71	413.53	446.55	2.28	53.134	O
2.37	+18.03	+84.11	+22.42	+26.72	+21.30	2.37	421.85	1758.12	858.90	413.50	444.30	2.37	54.085	O
2.45	+18.31	+84.29	+21.31	+26.53	+21.76	2.45	424.07	1753.98	857.78	413.75	445.31	2.45	53.926	O
2.53	+18.88	+84.65	+20.80	+25.29	+22.40	2.53	431.28	1772.70	862.48	422.27	451.79	2.53	53.947	O
2.62	+19.63	+84.93	+20.88	+16.98	+23.24	2.62	431.11	1766.39	866.30	423.97	452.55	2.62	53.117	O
2.70	+19.64	+85.04	+20.38	+4.32	+23.30	2.70	435.99	1794.40	872.30	428.94	458.22	2.70	53.086	O
2.78	+20.24	+85.54	+20.59	+3.63	+23.76	2.78	436.20	1796.18	875.24	427.61	454.97	2.78	53.103	O
2.87	+20.62	+85.76	+20.58	+3.15	+24.01	2.87	433.73	1782.99	876.22	425.24	454.69	2.87	54.275	O
2.95	+20.49	+85.61	+20.19	+2.50	+23.77	2.95	425.70	1773.06	873.62	417.79	445.65	2.95	53.009	O
3.03	+20.98	+85.80	+20.03	+2.18	+23.92	3.03	424.39	1763.30	869.69	417.45	445.74	3.03	53.041	O
3.12	+21.31	+86.24	+20.02	+1.99	+24.26	3.12	419.56	1751.62	864.59	412.50	441.88	3.12	53.988	O
3.20	+21.80	+86.67	+20.07	+1.99	+24.68	3.20	419.45	1741.77	859.95	411.40	438.19	3.20	53.072	O
3.28	+20.80	+86.85	+19.63	+2.07	+24.27	3.28	430.32	1753.28	861.47	423.79	452.36	3.28	52.273	O
3.37	+22.45	+87.67	+20.21	+3.18	+24.75	3.37	436.94	1779.31	866.41	429.51	456.40	3.37	52.940	O
3.45	+21.94	+87.35	+19.77	+3.09	+24.08	3.45	442.30	1796.62	872.64	435.30	463.62	3.45	53.960	O
3.53	+22.64	+87.88	+20.28	+3.77	+24.25	3.53	450.20	1813.22	878.43	444.25	471.86	3.53	54.061	O
3.62	+22.29	+87.84	+20.02	+3.51	+23.47	3.62	448.40	1817.00	886.30	441.58	466.75	3.62	54.946	O
3.70	+21.82	+87.79	+19.92	+3.37	+22.68	3.70	440.43	1812.59	893.19	435.50	464.84	3.70	54.348	O
3.78	+21.97	+88.19	+20.09	+3.30	+22.06	3.78	440.03	1806.53	894.36	433.30	459.14	3.78	54.777	O
3.87	+21.72	+88.30	+20.11	+3.13	+21.04	3.87	436.75	1795.63	897.31	430.42	458.47	3.87	53.992	O
3.95	+21.74	+88.67	+20.20	+3.05	+20.33	3.95	430.44	1790.78	897.28	424.00	450.48	3.95	53.988	O
4.03	+21.60	+89.07	+20.49	+3.01	+19.28	4.03	427.80	1774.10	895.63	420.63	448.81	4.03	53.079	O
4.12	+21.12	+89.18	+20.46	+2.61	+18.12	4.12	421.68	1763.57	893.48	416.47	444.96	4.12	53.113	O
4.20	+20.67	+89.22	+20.32	+2.49	+16.77	4.20	430.59	1763.61	890.40	424.18	451.03	4.20	52.256	O
4.28	+19.65	+89.19	+20.06	+2.21	+15.27	4.28	426.41	1761.72	891.00	420.28	447.80	4.28	52.145	O
4.37	+19.49	+89.38	+20.01	+2.09	+13.59	4.37	422.90	1752.23	890.24	417.23	444.71	4.37	52.031	O
4.45	+19.25	+89.92	+20.45	+2.47	+4.75	4.45	423.82	1753.18	889.64	416.99	445.27	4.45	52.328	O
4.53	+18.58	+89.95	+20.32	+2.34	+2.69	4.53	423.61	1758.22	889.20	417.14	446.32	4.53	52.256	O
4.62	+18.06	+90.16	+20.51	+2.45	+2.38	4.62	425.50	1759.91	889.20	418.24	446.30	4.62	52.238	O
4.70	+17.31	+90.29	+20.29	+2.27	+2.13	4.70	424.75	1764.64	889.99	417.41	448.69	4.70	52.280	O
4.78	+16.94	+90.53	+20.59	+2.50	+2.33	4.78	424.45	1767.96	890.90	416.38	447.53	4.78	53.086	O
4.87	+16.04	+90.34	+20.29	+2.30	+1.95	4.87	422.67	1764.42	892.29	413.98	446.01	4.87	53.169	O
4.95	+15.54	+90.61	+20.40	+2.32	+1.88	4.95	421.77	1758.25	891.20	414.49	446.88	4.95	53.106	O
5.03	+14.88	+90.60	+20.32	+2.17	+1.79	5.03	422.07	1764.40	892.09	413.36	446.47	5.03	53.182	O
5.12	+14.23	+90.63	+20.16	+1.84	+1.38	5.12	420.48	1757.43	891.05	411.09	446.96	5.12	53.099	O
5.20	+13.92	+90.66	+20.27	+1.82	+1.44	5.20	418.22	1750.53	889.42	408.51	443.54	5.20	53.082	O
5.28	+13.65	+90.75	+20.33	+1.77	+1.43	5.28	416.97	1749.36	889.68	408.43	445.68	5.28	53.120	O
5.37	+13.64	+91.01	+20.53	+2.02	+1.54	5.37	418.06	1744.00	887.52	407.56	445.27	5.37	53.106	O
5.45	+13.43	+91.01	+20.57	+2.03	+1.54	5.45	416.68	1742.18	886.80	406.78	443.38	5.45	52.228	O

Sample Test Data File

5.53	+13.09	+91.11	+20.37	+1.70	+1.27	5.53	416.41	1744.86	884.84	406.24	442.95	5.53	52.238	O
5.62	+13.12	+91.25	+20.48	+1.73	+1.44	5.62	416.20	1737.63	884.35	405.16	441.41	5.62	52.228	O
5.70	+13.16	+91.40	+20.54	+1.89	+1.78	5.70	418.20	1741.39	884.10	404.89	443.62	5.70	52.204	O
5.78	+12.79	+91.23	+20.32	+1.79	+1.86	5.78	419.19	1748.29	882.21	405.19	441.81	5.78	52.193	O
5.87	+13.23	+91.50	+20.56	+2.11	+2.97	5.87	418.53	1736.87	883.70	406.06	445.89	5.87	53.096	O
5.95	+13.17	+91.64	+20.53	+1.99	+3.61	5.95	418.14	1730.51	882.08	404.62	442.27	5.95	53.110	O
6.03	+13.29	+91.50	+20.57	+1.96	+4.36	6.03	418.08	1724.26	879.79	405.67	442.02	6.03	53.013	O
6.12	+13.46	+91.72	+20.71	+2.16	+4.97	6.12	418.39	1724.02	879.85	405.55	443.72	6.12	53.130	O
6.20	+13.48	+91.71	+20.58	+1.91	+5.57	6.20	419.74	1725.90	880.47	407.15	443.15	6.20	53.151	O
6.28	+13.72	+92.08	+20.73	+1.83	+6.20	6.28	418.65	1716.35	880.84	407.36	445.09	6.28	53.082	O
6.37	+13.86	+92.03	+20.70	+1.83	+6.77	6.37	419.92	1714.78	881.47	406.45	442.98	6.37	53.089	O
6.45	+13.91	+92.05	+20.63	+1.77	+7.21	6.45	420.11	1717.29	879.10	407.37	443.74	6.45	53.099	O
6.53	+13.75	+91.83	+20.33	+1.46	+7.25	6.53	419.83	1718.48	879.56	407.61	444.04	6.53	52.287	O
6.62	+14.13	+92.09	+20.65	+1.89	+7.98	6.62	421.47	1719.14	879.88	409.40	443.81	6.62	52.511	O
6.70	+14.11	+91.96	+20.40	+1.53	+8.13	6.70	422.01	1720.05	880.46	409.38	444.84	6.70	53.086	O
6.78	+14.40	+92.23	+20.55	+1.82	+8.54	6.78	422.10	1717.59	879.81	410.15	445.72	6.78	53.096	O
6.87	+14.50	+92.25	+20.62	+2.05	+8.84	6.87	421.57	1720.75	881.05	409.46	444.15	6.87	53.072	O
6.95	+14.56	+92.29	+20.50	+1.88	+8.86	6.95	424.20	1728.60	881.54	411.46	446.83	6.95	53.061	O
7.03	+14.58	+92.37	+20.50	+1.87	+9.02	7.03	425.18	1720.22	881.66	410.06	444.38	7.03	53.151	O
7.12	+14.56	+92.33	+20.38	+1.89	+9.16	7.12	424.23	1725.23	882.36	411.22	445.30	7.12	52.273	O
7.20	+14.85	+92.41	+20.66	+2.24	+9.44	7.20	422.41	1727.22	882.76	411.94	446.39	7.20	52.224	O
7.28	+14.80	+92.54	+20.63	+2.17	+9.40	7.28	425.42	1725.72	883.28	411.39	444.15	7.28	52.231	O
7.37	+14.74	+92.49	+20.47	+2.10	+9.28	7.37	424.73	1726.48	885.02	412.69	448.19	7.37	53.162	O
7.45	+14.80	+92.61	+20.54	+2.23	+9.42	7.45	426.13	1729.78	882.96	413.15	445.99	7.45	53.130	O
7.53	+15.04	+92.94	+20.81	+2.58	+9.50	7.53	425.75	1725.25	884.79	413.37	446.76	7.53	52.211	O
7.62	+14.89	+92.90	+20.65	+2.44	+9.23	7.62	424.83	1732.59	885.83	413.60	447.21	7.62	53.079	O
7.70	+14.69	+92.77	+20.50	+2.25	+8.90	7.70	425.55	1731.00	886.39	411.93	445.72	7.70	53.130	O
7.78	+14.74	+92.99	+20.71	+2.43	+8.88	7.78	425.23	1734.32	887.06	413.46	445.77	7.78	52.252	O
7.87	+14.68	+92.97	+20.66	+2.46	+8.57	7.87	427.73	1736.15	887.52	412.71	445.70	7.87	52.273	O
7.95	+13.91	+92.84	+20.44	+2.13	+7.99	7.95	425.67	1732.42	888.72	413.01	444.31	7.95	51.453	O
8.03	+14.56	+93.14	+20.67	+2.37	+7.79	8.03	424.18	1731.80	887.68	412.00	446.16	8.03	52.425	O
8.12	+14.56	+93.39	+20.82	+2.51	+6.88	8.12	424.30	1729.85	888.73	412.06	445.24	8.12	52.034	O
8.20	+13.97	+92.94	+20.52	+2.10	+3.55	8.20	425.07	1735.32	889.18	412.98	444.57	8.20	52.207	O
8.28	+14.42	+93.37	+20.85	+2.52	+3.45	8.28	424.62	1739.02	889.52	412.17	446.16	8.28	52.017	O
8.37	+14.17	+93.36	+20.75	+2.25	+2.47	8.37	423.89	1730.99	890.45	411.46	445.12	8.37	52.200	O
8.45	+13.71	+93.29	+20.56	+2.11	+2.16	8.45	422.89	1730.77	890.33	411.64	443.76	8.45	52.221	O
8.53	+13.90	+93.40	+20.70	+2.16	+2.21	8.53	423.84	1733.51	890.73	411.59	445.91	8.53	52.494	O
8.62	+13.96	+93.69	+20.88	+2.48	+2.23	8.62	424.16	1734.24	891.33	411.92	444.49	8.62	53.051	O
8.70	+13.85	+93.65	+20.85	+2.41	+2.24	8.70	422.68	1731.61	890.23	411.29	444.96	8.70	53.016	O
8.78	+13.75	+93.75	+20.89	+2.44	+2.24	8.78	421.98	1731.09	891.63	409.60	444.20	8.78	53.099	O
8.88	+13.69	+93.76	+20.91	+2.37	+2.09	8.88	424.48	1726.13	891.01	410.92	444.20	8.88	52.985	O
8.95	+13.50	+93.63	+20.71	+2.17	+1.95	8.95	423.21	1730.95	891.40	410.76	446.44	8.95	53.248	O
9.03	+13.52	+93.72	+20.87	+2.34	+1.97	9.03	423.38	1726.55	891.69	409.70	443.62	9.03	53.099	O
9.12	+13.45	+93.84	+20.79	+2.27	+1.94	9.12	421.45	1729.20	891.00	410.06	445.22	9.12	53.120	O
9.20	+13.26	+93.71	+20.55	+2.09	+1.94	9.20	422.13	1728.73	890.01	409.38	444.79	9.20	53.099	O
9.29	+13.40	+93.88	+20.80	+2.16	+2.07	9.29	422.37	1729.35	890.44	409.64	442.52	9.29	53.137	O
9.37	+13.16	+93.98	+20.61	+1.99	+1.83	9.37	422.82	1728.21	890.75	410.08	445.19	9.37	53.117	O
9.46	+13.08	+93.88	+20.57	+2.02	+1.78	9.46	422.97	1726.10	890.05	410.44	445.15	9.46	53.075	O
9.54	+13.21	+93.98	+20.65	+2.17	+2.07	9.54	422.28	1727.54	891.19	410.06	444.22	9.54	53.034	O
9.62	+13.49	+94.17	+20.98	+2.41	+2.48	9.62	420.83	1725.62	889.61	409.32	445.91	9.62	53.051	O
9.70	+12.93	+93.89	+20.51	+2.04	+2.01	9.70	421.93	1725.63	889.98	409.52	443.32	9.70	52.816	O
9.79	+13.33	+94.22	+20.81	+2.40	+2.33	9.79	422.32	1722.11	889.31	409.29	444.49	9.79	53.563	O
9.87	+13.39	+94.31	+20.87	+2.48	+2.57	9.87	423.68	1725.59	887.95	409.93	443.37	9.87	52.349	O
9.95	+13.12	+93.94	+20.55	+2.07	+2.40	9.95	421.95	1728.16	889.94	409.38	444.32	9.95	53.186	O
10.05	+13.11	+94.00	+20.55	+2.03	+2.56	10.05	423.27	1729.35	890.72	409.81	446.31	10.05	52.252	O
10.13	+13.35	+94.19	+20.83	+2.27	+3.06	10.13	422.50	1726.23	889.52	408.50	443.90	10.13	52.190	O
10.21	+13.18	+94.22	+20.59	+1.98	+2.75	10.21	421.60	1724.76	889.12	409.35	444.45	10.21	52.245	O
10.28	+13.35	+94.28	+20.82	+2.35	+2.80	10.28	422.10	1727.91	890.21	409.44	444.62	10.28	52.411	O
10.37	+13.37	+94.56	+20.81	+2.39	+2.90	10.37	422.50	1727.95	889.01	409.55	443.72	10.37	52.321	O
10.45	+13.16	+94.40	+20.51	+2.08	+2.84	10.45	422.98	1730.98	889.01	410.54	445.78	10.45	52.052	O
10.53	+13.40	+94.53	+20.82	+2.41	+2.80	10.53	423.59	1727.05	889.85	410.30	443.86	10.53	52.231	O
10.63	+13.17	+94.25	+20.57	+2.11	+2.17	10.63	423.16	1731.35	889.53	410.73	445.08	10.63	52.283	O
10.70	+13.16	+94.34	+20.60	+2.28	+2.37	10.70	423.18	1730.55	887.77	410.28	443.86	10.70	52.211	O
10.78	+13.33	+94.60	+20.73	+2.47	+2.37	10.78	423.71	1727.57	890.68	411.08	444.48	10.78	52.197	O
10.87	+13.16	+94.54	+20.61	+2.21	+2.14	10.87	422.67	1733.76	890.14	411.66	445.21	10.87	53.034	O
10.95	+13.18	+94.50	+20.69	+2.15	+2.11	10.95	424.43	1734.97	890.57	410.64	444.28	10.95	52.435	O
11.03	+13.05	+94.32	+20.58	+2.08	+2.05	11.03	422.68	1735.46	889.98	410.32	446.25	11.03	52.314	O
11.12	+13.33	+94.79	+20.87	+2.24	+2.25	11.12	424.24	1733.28	889.86	411.32	443.62	11.12	52.224	O
11.20	+13.09	+94.60	+20.59	+2.17	+1.96	11.20	423.83	1732.11	891.92	411.43	446.81	11.20	53.103	O
11.28	+12.88	+94.34	+20.49	+2.07	+1.93	11.28	421.64	1735.03	890.64	410.20	444.74	11.28	53.082	O
11.37	+12.75	+94.30	+20.40	+2.00	+1.81	11.37	426.05	1731.61	891.21	410.89	443.84	11.37	53.694	O